International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-4, November 2019

Optimization of the ANOVA Procedure for Support Vector Machines

Borislava Vrigazova, Ivan Ivanov

Abstract*: Feature selection is a powerful tool to identify the important characteristics of data for prediction. Feature selection, therefore, can be a tool for avoiding overfitting, improving prediction accuracy and reducing execution time. The applications of feature selection procedures are particularly important in Support vector machines, which is used for prediction in large datasets. The larger the dataset, the more computationally exhaustive and challenging it is to build a predictive model using the support vector classifier. This paper investigates how the feature selection approach based on the analysis of variance (ANOVA) can be optimized for Support Vector Machines (SVMs) to improve its execution time and accuracy. We introduce new conditions on the SVMs prior to running the ANOVA to optimize the performance of the support vector classifier. We also establish the bootstrap procedure as alternative to cross validation to perform model selection. We run our experiments using popular datasets and compare our results to existing modifications of SVMs with feature selection procedure. We propose a number of ANOVA-SVM modifications which are simple to perform, while at the same time, boost significantly the accuracy and computing time of the SVMs in comparison to existing methods like the Mixed Integer Linear Feature Selection approach.

***Keywords*: support vector machines, ANOVA, bootstrapping, PCA transformation, feature selection**

I. INTRODUCTION

The support Vector Machines (SVMs) was first introduced by Vapnik and Lerner [27] and further developed by Vapnik and Chervonenski [26]. The baseline version of SVM was developed by Cortes and Vapnik [5] and implemented in the LIBSVM library [4]. Since then, many modifications of the SVMs have been used to improve its performance. Big part of them is widely applied in biology [10], [16] and [17], due to their advantages summarized by Vapnik [27] and Yu [30]. The aim of our paper is to propose optimized versions of the SVMs based on the analysis of variance that can be applied to various types of data.

Hsu and Lin [6] proposed two versions of the SVMs for multiclass problems as the support vector classifier was originally developed for binary classification. They compared their results to similar methods described in Weston's paper [28]. Current versions of SVMs can solve multiclass optimization problems, some of which are based on feature selection methods [7]. Bradley [2] discussed the advantages of feature selection in the support vector classifier. Weston summarized popular feature selection methods for SVMs in [29].

Mangasarian [14] proposed a feature selection approach for nonlinear kernel SVMs. In the same year, Uncu [24] combined wrapper methods and filters to identify the relevant features in SVMs. In 2009 Maldonado [13] proposed a novel wrapper SVMs. The mechanism behind wrapper methods and filters is similar to ranking the importance of features via ANOVA [1], [30]. Reduced support vector machines is another version based on feature selection [9], [11]. It can be combined with clustering for improved performance [8]. Feature selection with L1-norm is another method for minimizing the dimension of features for SVMs [15], [19] and [20] similar to the least shrinkage operator. Neumann [18] and Rakotomamonjy [21] proposed modifications of SVMs based on regularisation and embedded nonlinear feature selection. Ngyuen [20] proposed a method for optimal feature selection in the SVMs. The mixed linear approach is also applied as a tool for feature selection in the SVMs [31]. Among its benefits are increased accuracy and stable choice of features compared to the ordinary SVMs [12]. A recent example is the research article of Maldonado [12], in which feature selection methods for the SVMs are extended using the Mixed Integer Linear Approach.

He based his paper on the l1-SVM formulation in [2] and the LP-SVM method in [31]. He modified the two methods to be applicable to the Mixed-Integer Programming framework. Maldonado proposed two modified versions of the l1-SVM formulation and LP-SVM method, which allow performing feature selection in one step, better accuracy compared to ranking methods at the cost of reasonable computing time. He compared his results to ranking feature selection methods and highlighted the practical advantages of his modifications. He performed his experiments in Matlab using a range of freely available datasets, including a dataset where the rule n > p was broken showing that his versions can provide stable practical solutions despite the irregular structure of data.

We further examine the feature selection problem in the context of Support vector machines, particularly how to select the optimal number of features to improve the accuracy and time of the model. We summarize our key findings on the topic by proposing a number of modifications of the ANOVA method for improving the performance of the SVMs.

Manuscript published on November 30, 2019.
* Correspondence Author
Borislava Vrigazova*, Department of Statistics and Econometrics, Sofia University, Sofia, Bulgaria. Email: vrigazova@uni-sofia.bg
Ivan Ivanov*, Department of Statistics and Econometrics, Sofia University, Sofia, Bulgaria. Email: i_ivanov@feb.uni-sofia.bg

We show that our proposed versions of ranking feature selection for SVMs can improve accuracy and computing time obtained by the Mixed Integer Linear Approach in [12]. Section 2 describes our modifications. Sections 3 reveals and discusses our results. Section 4 concludes.

II. METHODOLOGY

A. The classical Support vector machines

First, we introduce the mechanism of the support vector classifier. Many software packages use the LIBSVM library [4] that was developed based on Vapnik's original paper [5]. The standard Support vector machines that is based on the LIBSVM library can be fitted using the scikitlearn.svm.SVC module in Python [22]. The mechanism behind the C-Support vector is given in [4] and described by eq.1:

$$min\frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\xi_i, \qquad C > 0 \tag{1}$$

Subject to:

$$y_i(w^T\phi(x_{i*}) + b) \geq 1 - \xi_i, \ where \tag{2}$$

$$\xi_i \ \geq 0, i = 1,..l$$

In the standard SVMs method x_{ij} denotes the matrix that contains the input training data. The matrix consists of x_{i*} observations, i=1,..l, and x_{*j} independent variables, j=1,..p. The parameter $\varphi(x_{i*})$ maps the kernel function applied to the training space of $x_{i*,}$ observations. The constant C, C > 0, is the regularization parameter that tunes the soft margin and controls the shrinking of SVM coefficients. The parameter *ksi* ξ_{ij} accounts for the error term. The weight vector w contains the weight of each variable as a result of the C optimization [5]. The routine procedure for solving the optimization problem in eq.1 is to standardize the input data by using eq. 3 and fit the Support vector machines:

$$z_{ij} = \frac{x_{ij-}\mu}{\sigma} \tag{3}$$

Where μ and σ denote the mean value and the standard deviation of the variable. The standard SVMs in eq. 1 is subject to eq. 2 and 3. In the standard SVM procedure, the value of the constant C is selected using cross validation [5]. The value of C that minimizes eq. 1 is selected and then used to fit the support vector classifier on dataset that is split into training and test set.

We present the equation of normalization (eq. 4) as we will use it in our modifications:

$$n_{ij} = \frac{z_{ij} - \min(z_{ij})}{\max(z_{ij}) - \min(z_{ij})} \tag{4}$$

In the next subsections we present our optimized versions of the C-optimization in the support vector classifier.

B. Modification 1: ANOVA-CV-L-SVM

1 We first standardize (eq.3) our input data by using the StandardScaler() function in Python. The standardization is proposed in [23]. This procedure removes the unit from the input data.

2 The new step that we introduce is normalization (eq.4) of the dataset so that all features have values between 0 and 1 using the MinMaxScaler() function in scikitlearn. In this way, we remove negative values from the standardized data.

3 We then fix C in eq. 1 to 1 instead of selecting it using cross validation as in [5]. Fixing C to 1 balances the trade-off between small errors and large margin.

4 We split the dataset into training and test set using the built-in tenfold cross validation function in scikit learn as described in (Stiston et al., 1997). The ratio we use is 70:30. We use this procedure to perform model selection.

5 We run the C- SVM classifier described in eq. 1 using the linear kernel described in [5]. In Python we used the function sklearn.svm.SVC. The input data are now subject to eq. 2, 3 and 4.

6 We fit the SVMs (eq. 1) subject to eq. 2, 3 and 4 for each percentile of features and keep as many features as needed to achieve the highest accuracy of the model based on the analysis of variance. We fit the ANOVA-SVM using the functions SelectPercentile, chi2 for the ANOVA and sklearn.svm.SVC for the SVMs.

7 We calculate accuracy, AUC score and computing time (CPU). We call the linear version of our baseline algorithm the ANOVA-CV-L-SVM.

C. Modification 2: ANOVA-Bootstrap-SVM

1 We first standardize (eq.3) our input data by using the StandardScaler() function similar to the previous algorithm.

2 We fix C=1.

3 We then normalize (eq.4) the dataset so that all features have values between 0 and 1 using the MinMaxScaler() function in scikitlearn. Unlike the standard SVM procedure (eq. 1) subject to eq. 2 and 3, our version is subject eq. 2,3 and 4.

4 Unlike the standard SVM algorithm, we split the transformed dataset into training and test set using the bootstrap procedure described in [3]. The ratio we use is 70:30. In this step, we changed the classic model selection procedure by introducing the bootstrap.

5 For each percentile of features we then apply the C-SVM classifier described in eq. 1 subject to eq. 2, 3 and 4 with a linear kernel. We fit the ANOVA-SVM using the functions SelectPercentile, chi2 for the ANOVA and sklearn.svm.SVC for the SVMs.

6 We select the set of variables that produce the highest accuracy. We call the algorithm the ANOVA-BOOT-L-SVM.

7 We calculate the accuracy, AUC score and computing time (CPU) for the two modifications and compare our results to other SVM modifications.

D. Modification 3: ANOVA-PCA-Bootstrap-SVM

1 Like in previous algorithms, we first standardize (eq.3) our input data by using the StandardScaler() function .

2 A new transformation we apply on the standardized data is the PCA transformation [32] using the sklearn decomposition.PCA module. After the transformation, the size of the feature space decreases to p-1.

3 We normalize (eq. 4) the transformed dataset so that all features have values between 0. Normalization is performed to remove negative values from the transformed dataset.

4 After we prepare the dataset, we split the dataset into training and test set using the bootstrap procedure described in [3]. For each percentile of features we solve the C- SVM optimization problem described in eq. 1. subject to the constraints defined in eq.2, the PCA transformation, eq. 3 and 4. We use a linear kernel to fit the model. We select the set of variables that produce the highest accuracy. We do that by using the functions SelectPercentile, chi2 for the ANOVA and sklearn.svm.SVC for the SVMs.

5 We call this algorithm the ANOVA-PCA-BOOT-L-SVM.

6 We calculate the accuracy, AUC score and computing time (CPU). We then compare our results to those described in [12] and the standard SVM.

III. RESULTS AND DISCUSSION

A. Datasets

We tested the performance of our algorithms on datasets that were used in [12], particularly breast cancer, Australia credit and pima diabetes datasets. All of them can be downloaded from the UCI Repository. We do not apply any prior transformations to data other than the ones described in the previous section. Table 1 presents our data:

Table I: Data sources

Dataset	Number of observations (l)	Number of features (p)	Source
Wisconsin breast cancer	569	30	[33]
Australian credit approval	690	14	[34]
PIMA diabetis dataset	768	8	[35]

B. Empirical results

Maldonado [12] proposed new versions of the Mixed integer linear approach for feature selection in Support vector machines that chose the optimal number of features for improving the accuracy of SVM. We compare our results to his. Table 2 makes comparison in terms of accuracy (ACC), AUC scores (AUC) and number of features selected (k). The results marked by asterisks can be reviewed in [12]. The rest of the table presents the performance of our modifications presented in the Methodology section and the classical SVM (eq. 1 subject to eq. 2 and 3). To find the value of C in the classical SVM, we used two grids of values for C: values between 0.00001 and 10 in the first grid and between 0.00001 and 100 in the second grid. We performed the model selection in the classical version of SVM using tenfold cross validation. Table 2 shows the results.

Table 2 shows that Maldonado's experiments resulted in very high accuracy of SVMs on the three dataset. He achieved best accuracy for the Australia credit dataset equal to 85.7%, AUC score 86.3% with only ten of fourteen variables. The ANOVA-CV-L-SVM model provides the lowest accuracy of all modifications (84.8%), while the bootstrapped version resulted in higher accuracy (86.1%) than all other modifications. The best performance of the SVMs we achieved was through the ANOVA-PCA-BOOT-SVM version (87.8% accuracy and 91.6% AUC score compared to 85.7% ACC and 86.3% AUC in MILP2). Our modification improved both the accuracy and the AUC score of SVM using only four of fourteen input variables. Table 3 shows that Maldonado's MILP2 method took approximately 0.3 seconds to run. In contrast, the ANOVA-PCA-BOOT-SVM took only 0.05 seconds to run. Our modification provided faster computing time, improved accuracy and AUC score due to decreased number of features that contain the most important characteristics of the Australia credit dataset.

Our modified ANOVA algorithms improved the classification performance on the PIMA dataset as well. The mixed linear integer approach (MILP2) reached maximum accuracy of 78%, AUC score 73.4 for 0.20s and 0.30s respectively (Table 3 - MILP2 NFS and MILP FS). We improved the SVMs' performance using the ANOVA-BOOT-SVM algorithm. This algorithm reached 79.7% accuracy and 82.2 AUC score in 0.15s. The ANOVA-BOOT-SVM algorithm outperformed Maldonado's SVM versions using only seven features in contrast to eight in his approach (tables 2 and 3).

The breast cancer dataset also provided similar results. The maximum accuracy we achieved via the ANOVA-CV-L-SVM is 97.8% compared to 98.1% via the MILP1 approach [12]. We achieved accuracy very close to Maldonado's approach but at a higher speed (0.10s compared to 0.20s in his case). Provided that the difference between the two approaches is small (0.3 p.p.) and the accuracy in both cases is higher than 97%, we consider the MILP1 algorithm and the ANOVA-CV-L-SVM algorithm to yield similar results. The ANOVA-CV-L-SVM modification, however, was faster than the MILP1 method. Our finding suggests that we would consider the ANOVA-CV-L-SVM as a better approach to fit SVM as it provided accuracy high enough and faster calculations.

A key reason for the behavior of our model is the small number of features. The ANOVA-CV-L-SVM used only twenty-one as compared to twenty-six in the MILP1 method.

Note that the accuracy in the MILP1 method of 98.1% may be as a result of using too many variables. We decreased the number of features in the model by three and the accuracy falls only by 0.3p.p., which is negligible. This is another reason why we consider our accuracy of 97.8% robust. Like in the previous datasets, the computational advantage of our algorithm is stressed.

Table- II: Best accuracy and AUC, in percentage, and number of selected features (k) for AUS, WBC, and PIMA datasets.

	Australia Credit			Breast Cancer			PIMA		
	ACC	AUC	k	ACC	AUC	k	ACC	AUC	k
l2-SVM*	85.7	86.3	14	97.9	97.3	30	77.9	73.3	8
LP-SVM*	85.7	86.3	14	97.2	96.5	30	77.9	73.3	8
l1-SVM*	85.5	86.2	12	97.5	97.2	10	77.9	73.3	8
Fisher+ SVM*	85.5	86.2	2	97.9	97.3	20	77.5	72.4	7
RFE-SVM*	85.5	86.2	2	97.9	97.3	23	77.1	71.8	3
l0-SVM*	85.5	86.2	2	97.9	97.3	16	77	71.8	5
MILP1*	85.5	86.2	2	**98.1**	**97.7**	**26**	77.9	73.3	8
MILP2*	85.7	86.3	10	97.9	97.3	17	78	73.4	8
ANOVA-CV-L-SVM	84.8	90.8	1	97.8	99.5	21	78.1	83.0	8
ANOVA-Boot-SVM	86.1	92.2	6	97.3	99.1	24	**79.7**	**82.2**	**7**
ANOVA-PCA-Boot-SVM	**87.8**	**91.6**	**4**	96.0	99.2	27	78.2	82.8	8
Classical SVM with linear kernel C=10	85.51	92.9	14	97.01	99.98	30	76.95	84.59	8
Classical SVM with linear kernel C=100	85.5	92.84	14	96.13	99.99	30	76.95	84.62	8

Source: * [12], authors' calculations

Table III a) Comparison of computing time – Maldonado's methods:

	AUS	WBC	PIMA
l2-SVM*	0.50	0.20	0.30
LP-SVM*	0.20	0.10	0.10
MILP1-NFS*	0.20	0.20	0.20
MILP2-NFS*	**0.20**	0.30	**0.20**
Fisher + SVM*	0.50	0.20	0.40
l1-SVM*	0.40	0.40	0.30
RFE-SVM*	0.80	0.40	0.60
l0-SVM*	0.80	0.50	1.30
MILP1-FS*	0.20	0.20	0.20
MILP2-FS*	**0.30**	0.20	**0.30**

Source: [12]

Table III b) Comparison of computing time – Classical SVM and modifications:

	AUS	WBC	PIMA
ANOVA-CV-L-SVM	0.04	0.04	0.10
ANOVA-Bootstrap-SVM	0.09	0.05	**0.15**
ANOVA-PCA-Bootstrap-SVM	**0.05**	0.07	0.08
Classical SVM with linear kernel C=10	4.81	0.24	2.21
Classical SVM with linear kernel C=100	41.30	1.25	14.34

Source: authors' calculations

We also calculated the value of the target function (eq. 1) for our proposed versions of SVMs and compared it to the value resulting from the standard SVM. We were interested to discover whether the value of the target function would be the smallest for the best model. We calculated the value of the target function only for the model that we identified as the best for a dataset. Table 4 presents the results.

Table IV: Value of target function for AUS, WBC, and PIMA datasets

	WBC	AUS	PIMA
ANOVA-CV-L-SVM	**3.81**	1.00	3.98
ANOVA-Bootstrap-SVM	3.93	4.60	**3.90**
ANOVA-PCA-Bootstrap-SVM	5.31	**2.95**	4.29
Classic SVM with linear kernel C=10	17.58	12.13	11.14
Classic SVM with linear kernel C=1000	122.74	102.18	101.14

Source: authors' calculations

Retrieval Number: D7375118419/2019©BEIESP
DOI:10.35940/ijrte.D7375.118419
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Table 4 shows that in the PIMA dataset, the ANOVA-BOOTSTRAP-SVM provided the best accuracy and it had the smallest value of the target function - 3.90. The same is the case with the breast cancer database. The method that provided the best accuracy for the Australia dataset is the ANOVA-PCA-BOOTSTRAP-SVM. Table 3 shows that the value of the target function for this modification is 2.95.

The value for the ANOVA-CV-L-SVM is one and it is the smallest. We can explain this outlier with the fact that in this case the target value is equal to the number of chosen features (one), which are not optimal for fitting SVMs. We confirm that finding by looking at the value of the weight vector w used in eq.3. The weight of that one variable chosen is zero, which means that the value of the target function in this case shows that only the intercept is fitted. As a result, this value of the target function reflects SVM model with only 1 significant variable. As this is a special case, we consider the ANOVA-PCA-BOOTSTRAP-SVM to have the smallest value of the target function. The analysis of table 3 validates our experiments by confirming that the lowest value of the target function when the number of features chosen is not one, corresponds to the best accuracy. Another important finding is that the lowest value of the target function in several ANOVA modifications for SVM corresponds to the smallest number of features chosen. Another finding we observe in our experiments is that our algorithms resulted not only in the highest accuracy at a reduced computing time, but also in the smallest number of optimal features compared to Maldonado's best model for each dataset.

A key problem in our experiments is how to choose the number of features in SVMs so that we affect the accuracy and the computing time of the model. A key finding from our experiments is that the fewer the features in the model, the faster is the computing time. Maldonado chooses ten features out of fourteen to fit MILP2 SVM on the Australia dataset, while we use only ten features. As a result, our ANOVA-PCA-BOOTSTRAP-SVM is faster. Similar is the case with the other two datasets - Maldonado [12] achieved highest performance on the breast cancer dataset using twenty-four features, while we used twenty-one and achieved robust accuracy of 97.8%. In the PIMA dataset he used all eight features, while we used only ten and our model shows better performance as discussed before.

Another important finding is that the classical SVM provided mediocre results on all three datasets compared to our suggestions. The aim of the standard SVM is to find the optimal value of the tuning parameter C so that the value of the target function in eq. 3 is minimized subject to eq. 2 and 3. This finding is confirmed in table 4 where we see the value of the target function defined in eq.3. The classical SVM did not minimize the target function compared to the other algorithms. Our modifications resulted in smaller value of the target function and the best modification we selected minimized it. To fit the classical SVM we used two grids of values for C: [0.00001:10] and [0.00001:100]. The procedure in our case outlined C=10 and C=100 as the best parameters to fit the classical version. However, the standard model's performance was worse than our modifications. The classical SVM was much slower and the accuracy was worse than our algorithms where we kept C = 1. Another disadvantage of the standard SVM is that there is not a standardized procedure for setting the grid of tuning parameters. The resulting output of the SVM becomes sensitive to the values in the grid. We solve this problem by fixing C=1 as it balances the trade-off between high margin and small error terms [22]. The overall performance of our algorithms was better than the standard SVM as we introduced new constraints on the input data, used different model selection procedures and fixed C to identify the important variables. This approach reduced the variables dimension and accelerated the computing time of the model.

In our modifications, we used cross validation similarly to Maldonado [12] but also introduced the bootstrap as a model selection method. The bootstrap model selection method introduced in the ANOVA-BOOT-SVM and ANOVA-PCA-BOOT-SVM splits the training and test set randomly but unlike cross validation, it uses a portion of the training and test data to perform model selection and allows for repetition [3]. As a result, the model selection can be performed on smaller dataset than in cross validation, which makes the operation faster. The bootstrap combined with the ANOVA method allows for fitting SVM faster, while the ANOVA selects the best number of features based on the smallest variance. Our ANOVA-CV-L-SVM also is an appropriate method for fitting SVM. Its advantage lies in avoiding overfitting, which combined with the analysis of variance method (ANOVA) chooses the optimal number of features that lowers the computing time and provides robust accuracy that is not inflated by the number of features.

C. Discussion

In this paper, we proposed several upgraded versions of the ANOVA-SVM that contribute to academic literature as they optimize the performance of the support vector machines. Among the advantages of our algorithms are simplicity, improved accuracy, reduced number of features and independence of the results from the value of the optimization as we keep it fixed to 1. In each of our algorithms we do not have to find the best value for C, so our results do not depend on C on each run. An important note should be made, however, that we fixed the value of C to 1 as this value balances the trade-off between high margin and small error terms. We did not experiment with values different from 1, so the execution time and accuracy of our proposed algorithms may differ in this case. Another important note is that Maldonado performed his research in Matlab, while we worked in Python. As the cores of our and Maldonado's algorithms are different, we believe the software environment does not play a key role in the execution time. Important role plays the algorithm that is executed. Therefore, we consider our and his results comparable from algorithmic point of view.

IV. CONCLUSION

In this paper, we showed that the performance of the ANOVA procedure for the support vector classifier depends very much on the conditions imposed on the optimization problem that the support vector machines solves. Imposing well-defined conditions can significantly boost execution time and improve the accuracy of the model.

Our findings suggest that the performance of the ANOVA-SVMs can further be boosted by using the bootstrap procedure for model selection instead of the tenfold cross validation. As a result, we enrich academic literature in the field by introducing SVM modified algorithms with practical advantages.

V. ACKNOWLEDGMENT

This research article has been supported by project N80-10-42/10.04.2019 under Sofia university's research programme.

REFERENCES

1. Bharathi A., Natarajan A.M., 2009, Minimal feature selection using SVM based on Anova, Bannari Amman Institute of Technology, Journal of Theoretical and Applied Information Technology
2. Bradley P., Mangasarian O., 1998, Feature selection vía concave minimization and support vector machines, in: Machine Learning proceedings of the Fifteenth International Conference (ICML'98) 82–90, San Francisco, California, Morgan Kaufmann
3. Breiman Leo, The Little Bootstrap and Other Methods for Dimensionality Selection in Regression: X-fixed Prediction Error, Journal of American Statistical association, 87, 738 -754
4. Chang C., Lin C., 2001, LIBSVM: A Library for Support Vector Machines, ACM Transactions on Intelligent Systems and Technology. 2 (3)
5. Cortes, C., and Vapnik V., 1995. Support-vector networks. Machine Learning, 20(3), 273–297.
6. Hsu, Ch., Lin Ch., 2002, A Comparison of Methods for Multiclass Support Vector Machines, IEEE Transactions on Neural Networks, vol. 13, no. 2, 415-425
7. Iannarilli F.J., Rubin P.A., 2003, Feature selection for multiclass discrimination via mixed-integer linear programming, IEEE Trans. Pattern Anal. Mach. Intell., 25, 779–783.
8. Jen L.-R., Lee Y.-J., Clustering model selection for reduced support vector machines, Proceedings of the Fifth International Conference on Intelligent Data Engineering and Automated Learning(IDEAL 2004), pages 714–719, Exeter, UK, 2004. Springer–Verlag.
9. Lee Y.-J., Mangasarian O. L., RSVM: Reduced support vector machines, Technical Report 00-07, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, July 2000. Proceedings of the First SIAM International Conference on Data Mining, Chicago, April 5-7, 2001, CD-ROM Proceedings. ftp://ftp.cs.wisc.edu/pub/dmi/techreports/00-07.ps
10. Lee, Y., C.K. Lee, 2003. Clasification of multiple cancer types by multicategory support vector machines using gene expression data. Bioinformatics, 19, 1132-1139
11. Lin K.-M., Lin C.-J., 2003, A study on reduced support vector machines, IEEE Transactions on Neural Networks, 14, 1449–1459
12. Maldonado S., Pérez J., Weber R., Labbé M., 2014, Feature selection for Support Vector Machines via Mixed Integer Linear Programming, Information Sciences, 279, 163–175
13. Maldonado S., Weber R., 2009, A wrapper method for feature selection using Support Vector Machines, Information Sciences 179, 2208–2217
14. Mangasarian O.L., Wild E.W., 2007, Feature selection for nonlinear kernel support vector machines, in: Seventh IEEE International Conference on Data Mining, IEEE, Omaha, NE, pp. 231–236
15. Mangasarian, O.L., 2007, Exact 1-Norm Support Vector Machines Via Unconstrained Convex Differentiable Minimization (Special Topic on Machine Learning and Optimization), Journal of Machine Learning Research, 7(2), 1517- 1530
16. Mingjun, S. , S. Rajasekaran, 2010. A greedy algorithm for gene selection based on SVM and correlation. Int. J. Bioinformat. Res. Appl., 6, 296-307.
17. Murat, C., M. Engin, E.Z. Engin and Y.Z. Atesci, 2009, Early prostate cancer diagnosis by using artificial neural networks and support vector machines. Expert Syst. Appl. Int. J., 36: 6357-6361. DOI: 10.1016/j.eswa.2008.08.010
18. Neumann, J., Schnorr, C. & Steidl, G., 2005, Combined SVM-based feature selection and classification, Machine Learning, 61(1), 129-150
19. Nguyen H., Franke K., Petrovi'c S., 2011, On General Definition of L1-norm Support Vector Machines for Feature Selection, International Journal of Machine Learning and Computing, Vol. 1, No. 3
20. Nguyen M., Torre F., 2010, Optimal feature selection for support vector machines, Pattern Recognition 43, 584–591
21. Rakotomamonjy, A., 2003, Variable selection using SVM based criteria. Journal of Machine Learning Research, 3:1357-1370
22. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011
23. Stitson M., Gammerman A., Vapnik V., Vovk V., Watkins C., Jason W., 1997, Support Vector Regression with ANOVA Decomposition Kernels, Technical Report CSD-TR-97-22
24. Uncu O., Türksen I.B., 2007, A novel feature selection approach: combining feature wrappers and filters, Inform. Sci. 177, 449–466.
25. Vapnik V., 2012, Statistical Learning Theory, John Wiley and Sons
26. Vapnik, V., and Chervonenkis A., 1964. A note on one class of perceptrons. Automation and Remote Control, 25.
27. Vapnik, V., and Lerner A., 1963. Pattern recognition using generalized portrait method. Automation and Remote Control, 24, 774–780.
28. Weston J., Watkins C., Verleysen M., 1999, Multi-class support vector machines,presented at the Proc. ESANN99, Brussels, Belgium,1999.
29. Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T. & Vapnik, V., 2001, Feature selection for SVMs. Advances in neural information processing systems, pp. 668-674.
30. Yu H., Kim J., Kim Y., Hwang S.,Lee Y.H., 2012, An efficient method for learning nonlinear ranking SVM functions, Inform. Sci. 209, 37–48
31. Zhou W., Zhang L., Jiao L. ,2002, Linear programming support vector machines, Pattern Recognition, 35, 2927–2936
32. Zou H., Hastie T., Tibshirani, 2006, Sparse principal component analysis, Journal of Computational and Graphical Statistics, 15(2): 262-28
33. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
34. http://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)
35. https://archive.ics.uci.edu/ml/support/diabetes

AUTHORS PROFILE

Borislava Vrigazova, Assistant professor Sofia university, https://www.researchgate.net/profile/Borislava_Vrigazova Borislava is currently a PhD candidate in Data science at Sofia University, Bulgaria. She obtained a master's degree in Statistics, financial econometrics and actuarial studies in 2015 after a bachelor's degree in Economics at the same university.

Her research areas include practical applications of machine learning algorithms for prediction and how their performance can be boosted. Also, applications of big data techniques to small datasets in the field of economics as alternative to traditional econometrics theory. She challenges traditional econometric modelling techniques used to find connections among variables from institutional economics by combining feature selection methods and big data prediction models. As a result, new applications of machine learning techniques to economic data appear.

prof. Ivan Ivanov, Dr.Science, **Sofia university,** https://www.researchgate.net/profile/Ivan_Ivanov20.

- Prof. Ivan Ivanov is Dr. Sc. Of Mathematical studies at Sofia University. Currently, he is head of the Data Science Laboratory at the Faculty of Economics and Business Administration, Sofia university. He is a co-founder of the master's degree in Business Analytics as well as the PhD program in Data Science, both at Sofia university, Bulgaria.

Prof. Ivanov has rich experience in the field of Applied Mathematics, having a numerous publication recognized by academics. His research interests in the field of machine learning are related to the optimization of algorithms' performance so that their practical advantages might be enhanced.