AI prediction model for Knee Arthroplasty

Computer Vision and Machine Learning

Source Code: github

Kaggle: mortenrosenquist

Morten Lyng Rosenquist
Faculty of Technical Sciences
Aarhus University
Aarhus, Denmark
201706031

April 23, 2022

Write Abstract

***Abstract*—**

***Index Terms*—**

Write keywords

I. Introduction

This paper will visit the development, training and evaluation of an AI prediction model. The model is to be used in a clinical setting regarding knee osteoarthritis treatment. A successful and effective treatment is the insertion of a Total Knee Arthroplasty (TKA). The number of TKA insertions are high in Denmark and it is expected to increase. TKA insertions are not always satisfactory and 5-10% will need revision within 10 years. The objective of the model is indivdualized patient selection to identify patient's risk group. This enables optimization before, under and after surgery, tailoring individual treatment plans and thereby increase the chance of the TKA surviving[3] .

The paper is done as a project in the course Computer Vision and Machine Learning at Aarhus University. The dataset is given together with an explanation of the features and description of the problem. The dataset is in advance split in training, validation and test sets. The training and validation observations are labeled. As the model is to determine wether a patient belongs in a risk group or not, it is a supervised binary classification problem.

A. Imbalanced Data

Building a model with imbalanced data can lead to problems. As the model primarily sees the majority class it might not learn enough from the minority class. This can lead to a model predicting all new observations as the majority class. This will lead to a high accuracy and recall for the majority class but a recall of 0 on the minority class. This is basically an useless model, that wont predict any of the minority class observations. In the clinical setting this means that we wont place any patients in the risk group. There are several ways to mitigate the issues with imbalanced data. The majority class can be downsampled, the minority class can be oversampled or for certain models we can tune the importance of each class during training [4].

B. Support Vector Machine

The model that will be used is a Support Vector Machine(SVM). It produces nonlinear boundaries by creating a linear boundary in a tranformed version of the feature space. With this functionality SVMs can be used for classification, regression and outlier detection. We will use it for classification. In its simplest form the SVM creates an linear optimal seperating hyperplane between two seperated classes. This is done by maximizing the margin between each support vector. Where classes can not be seperated by a linear boundary other kernels(polynomial, Radial Basis Function) can be utilized. There are hyper-parameters to be tuned of the SVM. The most important is the regularization parameter that determines the amount of punishment for misclassifications. Utilizing a polynomial kernel the degree of the kernel function is also to be tuned [2]. Default SVM does not perform well on imbalanced data, as the margin will favor the majority class. However, SVM can be extended to account for the importance of each class when maximizing the margin. The modified model is also reffered to as weighted SVM [4].

C. Feature Selection

The dataset contains a range of features regarding the patients. Having data with a high dimensionality can lead to problems such as long training time, a complex model and overfitting. Therefore, it will prefered if we can reduce the dimensionality while still having good performance of the model. This can be done by selecting the features that contribute the most to the target variable. This is typically supervised and will keep features intact. Principal Component Analysis can also be used to reduce the amount of features. This will however transform the features based on variance.

In regards to a clinical settings it would be interesting to see which features are the most valuable in terms of determining wether an patient is in the risk group.

II. Methods

A. Metrics

B. SVM

C. CV

Being stratified means that we have the same class ratio across all folds.

D. Imbalanced data

III. Experiments & Results

This section will cover the performed experiments, how they are conducted and their results. Initially, it will cover data preprocessing. Then the tuning of the model is described and assessed. The experiments are conducted in *python* using a *jupyter* notebook. The source code is to be found on github for potential replication. Additionally, the dataset used can be found on kaggle. Several python libraries are used to realize the experiments. Scikit Learn (sklearn)[1] is the primary library used for Machine Learning functionality.

A. Preprocesssing

The dataset contains two types of features - boolean (yes/no) and continous values (numeric). The dataset is not complete, meaning some observations are missing values in certain features. To overcome this the mean value is used for the numeric features and the most popular for the boolean features. As the features are in different units it is important to scale the data. Without scaling certain features might dominate the other features. This is specially the case for SVC, because it tries to maximize the distance between the support vectors and the hyperplane. *sklearn* have different options for scaling; StandardScaler, MinMaxScaler and RobustScaler. The MinMaxScaler with default settings will suffice with a scaling of each feature individually between 0 and 1[1].

B. Model training

Tuning the model to perform as desired is not a trivial task and it is highly dependent on the use case. Initially, there is several questions to be answered such as: How do we train the model? Do we use the raw training data? Are we to use sampling methods? How do we test different model parameters? Which metrics are used to train/evaluate the model? Answering these differently will lead to very diverse models. Due to lack of experience, this section will cover a trial and error approach. Firstly, a model is trained with default parameters. Then we look into tuning the model in various ways. The different models are concurrently evaluated.

sklearn's SVC implementation is used to build the model. Initially we try the model with the default parameters:

- Regularization parameter (C): 1
- Kernel: Radial Basis Function (rbf)

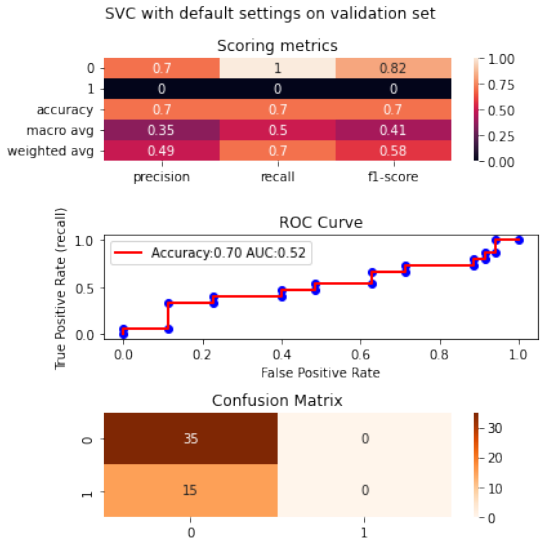

Fig. 1: Performance of SVC model with default parameters

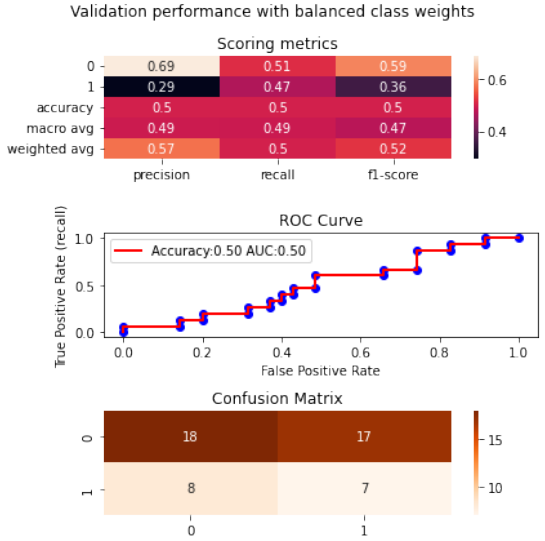

Fig. 2: Performance of SVC model with balanced class weights

The model is trained and evaluated. The evaluation on the validation set can be seen on Figure 1.

It is seen that the model scores well in terms of precision, recall and f1-score on class *0*. However, as seen in the confusion matrix this is due to the model classifying all observations to class *0*. This is due to the concerns introduced earlier regarding imbalanced data. As a measure, we balance the weights of each class. This is done by settings the model's *class_weight* parameter to *balanced*. *class_weights* can be defined explicitly or have *sklearn* calculate them by the proportions. The validation performance can be seen on Figure 2.

The update in class weights clearly makes the model predict more observations to class 1. This leads to better scoring

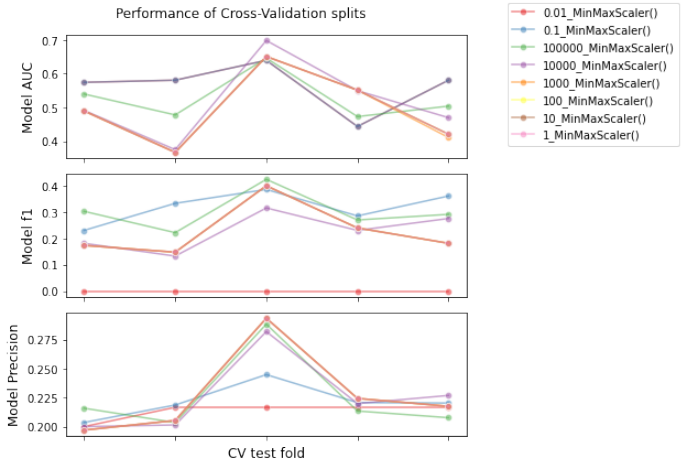

Fig. 3: Cross-validation performance of SVC model with different C values

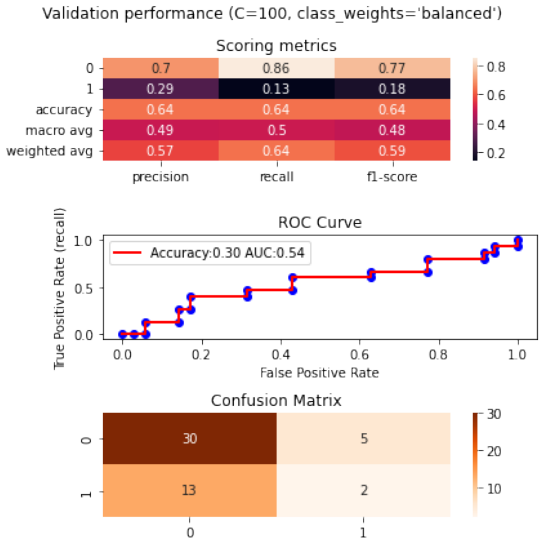

Fig. 4: Performance of best model looking at *C*

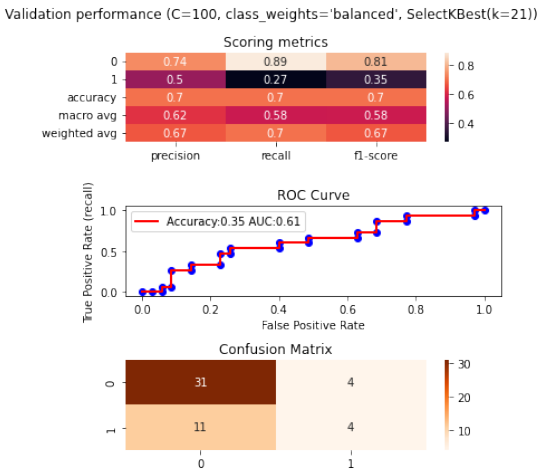

Fig. 5: Performance of best model looking at *C* and SelectKBest

of class 1, but many false positives are introduced as a consequence. To mitigate these, we try to tune other parameters of the model. We do this with *sklearn GridSearchCV* implementation. Here we can define a grid of parameters and train models with each combination of the parameters. Additionally, we utilize stratified cross validation with 5 folds. We define a grid with different options for the regularization parameter, C. The performance of the models on each folds can be seen on Figure 3.

There are eight different *C* values. However, less are showed on the plots. This is due to models performing similarly with nearby values. Looking at the different metrics; AUC, f1, precision, it can be seen that different C values affect the models. Additionally, models performing well at one metric might not succeed according to another metric. The best model is selected according to the average precision metric. The validation performance is evaluated. On figure 4 it shows, that we have less false positives and true positives.

There are still many misclassified observations. The next step that we will is feature selection. We will utilize *SelectKBest* with *chi2* as selection method. The selection step is included in the grid search of the best set of parameters. We try with a range of *k* features. The best model according to precision ends up being, *k=21, C=100*. The performance of this model can be seen on Figure 5.

The model seems to identify less false positives while classifying more true positives. Hence, selecting 21 features seems to be positive for the model. Next we will try different kernels of the SVC model. It showed that the default kernel that was utilized already, Radial Basis Functions, is the best performer according to precision.

C. Kaggle Submission

The final model is then used to predict labels on the unlabeled test data. The predictions are then submitted to kaggle and evaluated. The result was a score of 0.70666. This score is low compared to others in the competition. The low score can have various causes, such as:

- The model is not well tuned
- The model is overfitted on the training data
- The data is not preprocessed sufficient
- SVM is not the right classifier

During the tuning of the model a low range of values was used in the grid search of the optimal parameters. This was due to the necessity of being able to train the model on a workstation in tolerable time. It would be optimal to try a larger range of values and thus more combinations.

It is possible that the modes is being overfitted. Meaning the model are learning the wrong concepts from random fluctuations in the training data, that are not present in the test data. Tuning the regularization parameter of SVM and feature selection are both measures against overfitting. However, these

might not be sufficient.

The data might not be sufficiently preprocessed before being used by the model. Data cleaning was performed in terms of filling missing values, scaling was done by transforming the data in a specified range and feature selection was tried. Having additional processing power would allow to test other combinations of scaling and dimensionality reduction techniques. Additionally, it would be interesting to perform some outlier detection. It might be possible that there is some noisy data in the training data. This would also help to avoid overfitting.

Lastly, it might be the case that SVM is not the right model for the problem. SVMs are in general versatile with the use of various kernels. However, as seen in the results of the paper it showed, that even with non linear kernels it was not able to create the right seperating hyperplane.

IV. Conclusion

This paper succeeded in developing an AI prediction model to determine if a patient belongs in a specific risk group regarding knee osteoarthritis. Given data was preprocessed by filling missing values, normalizing and feature selection. Different versions of the SVM classifier was evaluated. Resulting in the optimal version having proportionality balanced class weights, a non-linear kernel and utilizing 21 of the original 23 features of the dataset. The performance of the resulting model was underwhelming in terms of precision. The source of the low performance is not clear. However, it might be caused by; model tuning, overfitting, data preprocessing or model selection.

For future work it would be interesting to look into aquiring more processing power to try out more versions of the model and different data preprocessing steps. This could include a larger range of values of the model's parameters, different scaling techniques and other techniques to fill out missing values in the data. Outlier detection could be used to identify potential noisy data. Additionally, other models than SVM should be investigated.

Write Conclusion

Other model

conclude in regards to clinical setting

References

[1] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[2] Jerome Friedman Trevor Hastie Robert Tibshirani. "The Elements of Statistical Learning - 2. Edition". In: Springer Series in Statistics. Springer, 2017. Chap. 12.

[3] Alexandros Iosifidis. "Kaggle_Problem_Description". In: (2022).

[4] Jason Brownlee. *Cost-Sensitive SVM for Imbalanced Classification*. URL: https://machinelearningmastery.com/cost-sensitive-svm-for-imbalanced-classification/. (accessed: 22.04.2022).