**APS 1052 Assignment 4**

**Peiyao Li**

**Beibei Zhang**

**Zhiqing Xu**

The answers to the questions are below in this document. All the code we used to answer those questions is in the Jupyter notebook file **Classification-SVM\_mod\_SPY\_complete.ipynb**. We have modified the code in other python files. To run the ipynb code, all the python scripts need to be unzipped. Some cells in the notebook are commented out since it takes a good while to run.

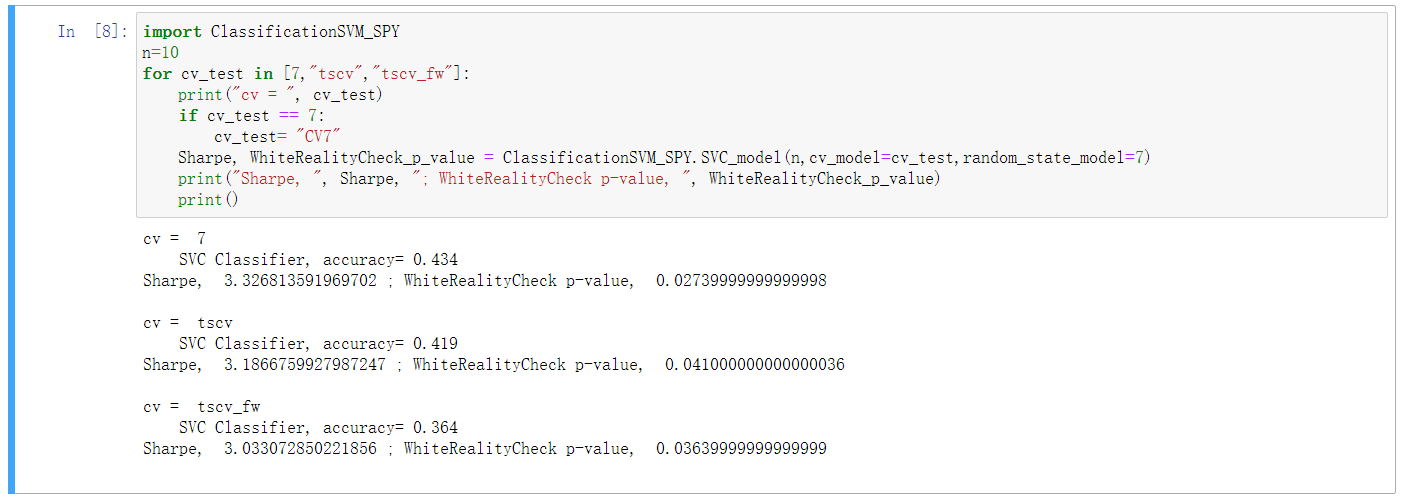
**Answer to Q1:**

No. It's not needed to add the linear kernel. According to data visualization, it is clear that the dataset is not linearly separable. Thus it requires a non-linear kernel such as rbf or polynommial.

## Answer to Q2:

When scoring = None, the estimator’s scoring method is used. In this case it is SVC's default scoring. SVC's scoring return the mean accuracy on the given test data and labels, which is used here. Different scoring measures can be used here, for example, "precision" or "roc\_auc", or multiple metrics can be evaluated at the same time by changing this value to a list of strings.

## Answer to Q3:



Models are training based on the three different ways of splitting the data, cv=7, tscv and tscv\_fw. Sharpe ratios are quite similar and WhiteRealityCheck p-value is low which implies strong statistical significance. The answer is YES, using different splitting methods will slightly change the result.

tscv\_fw and tscv are both timeseries split. tscv\_fw has fixed window of training set. It does not contain any overlapping folds at all! On the other hand, tscv has a growing window. The growing window split overlap at the beginning so the dataset in the earlier series are given more significance. This makes the difference in the final result.

When cv inputs are integer/None, and the estimator is a classifier, StratifiedKFold is used, which ensures the rearranging of data in each fold is a good representative of the whole. This will not retain the order of the time series data. It actually ruins the trend of the timeseries data, but sometimes would train a model with better predictions.

In theory, a normal K-fold cross-validation procedure is sufficient for machine learning models, with the assumption that the residuals of the model are uncorrelated. Theoretical proof was published in (Bergmeira, C. 2017 et. al). The author also stated that in most cases, cross-validation can effectively control overfitting. Even if the lag structure is not perfect, a normal K-fold cross-validation without any modifications can be used as a empirical solution. Therefore, we actually observe a better result when using cv=7 (split method without modifying for time-series).

## Answer to Q4:

The advantage of the fixed window is that a new set of training data and validation data is used for every training iteration. For growing window, the second time you train, you will include the dataset from the first training set, and the third time you train, you will have included the first training dataset three times, and the second training set two times. So the issue with growing window is the earlier dataset would have much more weighs contributed to the model since all the training set overlap at the beginning of the series. Whereas in the fixed window split, each data point contributes equally to the model output.

## Answer to Q5:(See ipynb for more details)

## 

A dummy classifier is used as a baseline of the model we trained. A comparison between the dummy classifier (which we used as a baseline) and the SVC model we trained allows us to concretely state the benefits of using this machine learning approach.

From the histogram that shows the size of the three signal classes (-1,0,1), we see the sizes are similar. Therefore, a dummy classifier that use a most\_frequent strategy (or use a "constant" strategy with constant being equal to the most frequent class) won't raise the accuracy of the prediction and is not favourable in this multi-class classification problem here.

A "stratified" dummy classifier strategy is to generate predictions by respecting the training set's class distribution, Since the sizes of three classes are very close to each other, using a "stratified" strategy and a "uniform" strategy are quite similar in this case. Since the class distribution of the training set and that of the test set are not necessarily similar, using a uniform strategy is more favourable here.

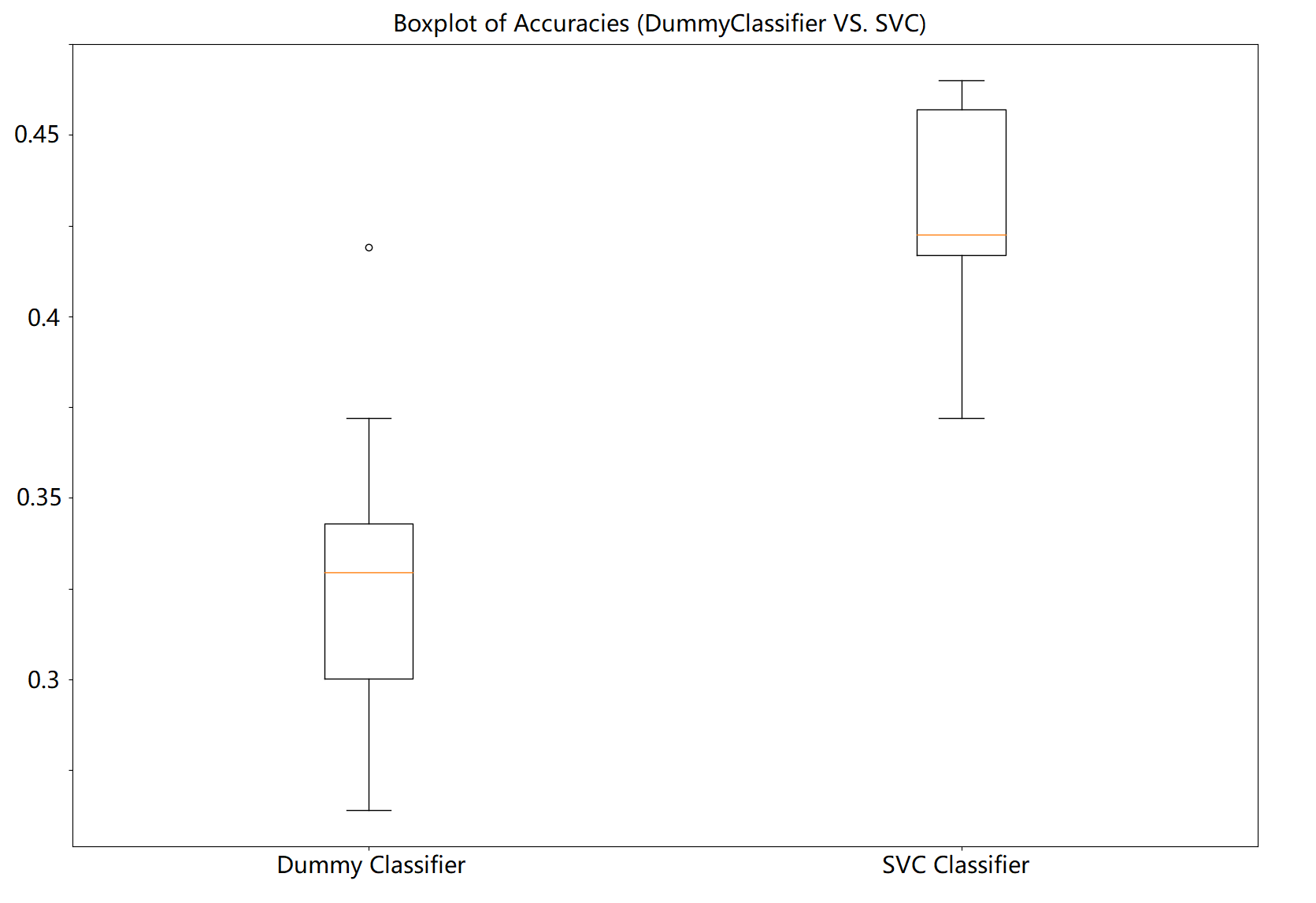
Without computing a confusion matrix, we can expect that using a "uniform" strategy for the dummy classifier will allow us to get a fairly good baseline that can be compared against the SVC model we trained to show an improved performance we achieved. As a matter of fact, this can be seen from the classification report and confusion matrix made for each dummy classifier strategy.

## Answer to Q6:

As discussed in Answer to Question 5, the dataset are fairly balanced as can be seen from the histogram (the size of each class is similar to each other). Therefore, the F1 score can be effectively ignored and the mis-classification rate is used for comparison between the classifiers here. Mis-classification rate is simply 1 - accuracy, the value of which can be obtained directly from the classification report printed in the output. In other words, we can simply compare the value of accuracy in the outputs. The accuracy of our classifier is 0.46 (random\_state=70) which is apparently larger than that of the dummy classifier which ranges from 0.26 to 0.37 (using a "uniform" strategy is to randonly generating predictions based on a uniform distribution, therefore the value of accuracy varies across multiple runs).

## Answer to Q7: (See ipynb for more details)

The value of random\_state are being set to 20 different values (0-19). Using the 20 values of random\_state, we obtain a list of accuracies of the svc classifier model. Although in some cases, our classifier model is not more than slightly better than the dummy classifier (i.e., random\_state=16), in most cases, we see a significant difference between the accuracy and the SVC classifier apparently has an improved performance of predicting the signals.



## Answer to Q8: (See ipynb for more details)

## 

As shown by the graph of the returns, the trading result of the SVC classifier is better than the baseline (that of the dummy classifier). The p-value being smaller than 0.05 shows the statistical significance of rejecting the null hypothesis which claims that the population distribution of rule returns has an expected value of zero or less. This implies the strategy based on SVC classifier actually brings profit not by chance or due to any trend in the data. On the other hand, the dummy classifier brings a significant loss. The large p-value also implies the same conclusion (the population distribution of rule returns has an expected value of zero or less).

## Answer to Q9:( See ipynb for more details)

## 

The length of the lookback window largely affects the result. The window setting equals to 10 seems to be the best one, based on the Sharpe ratio as well as WhiteRealityCheck p-value output by the above code. This result agrees with the PACF plot above, which shows lag 10 has a strong partial autocorrelation.

## Answer to Q10:

The metric currently used pays great attention on predicting the small changes rather than predicting the big changes. In order to get the big moves right, we want to model the big ups and downs more accurately but not the small ones. Therefore, we need to think of a way for the model to do that.

Another problem of the current metric is that it only reports the accuracy of the direction, rather than the magnitude, which is also very important when assessing the model. We can also try metric scoring that can reflect the magnitude, for example, instead of modeling a signal classifier, we can use a regression tree to take into account the magnitude directly. Another potential solution could be using an ensemble of models (i.e., Bayesian Ensemble Methods). Bayesian model averaging (BMA) is generally considered the standard model for creating ensembles of learners using Bayesian methods. Bayesian Model Combination (BMC) which is based on BMA, helps to select the best combination of models used for ensembles of learners. The potential learners we can use in an ensemble that models the magnitude include random forest, regression tree, support vector regressor, kernel ridge regressor, K-NN regressor, gaussian process regressor, gradient boosting regressor, etc. In this way, big ups and downs can be modelled by those learners. Ensembles can then be created using BMC methods to allow the model to improve its ability of predicting “big changes”.