folds): $\mathcal{D}_i^{\text{train}}$ for $i \in \{1,2,3,4,5\}$. Then, $\mathcal{M}_{\theta_n}$ is fitted on $\cup_{i\in\{1,2,3,4\}}\mathcal{D}_i^{\text{train}}$ (training set) and evaluated on $\mathcal{D}_5^{\text{train}}$ (validation set), let $\widehat{MSE}_5$ denote mean squared error of this model on $\mathcal{D}_5^{\text{train}}$. Afterwards, the same model is fitted again on $\cup_{i\in\{1,2,3,5\}}\mathcal{D}_i^{\text{train}}$, evaluated on $\mathcal{D}_4^{\text{train}}$ and leads to another error metric $\widehat{MSE}_4$. The same procedure can be repeated for five times with different validation set and creates five mean squared error metrics. Let $\overline{MSE}_{\theta_n}$ denote the average of $\widehat{MSE}_1$ to $\widehat{MSE}_5$ using hyper-parameter θ_n. Then, $\overline{MSE}_{\theta_n}$ constitutes an estimated test-time performance of model $\mathcal{M}_{\theta_n}$ on $\mathcal{D}^{\text{test}}$ when it is fitted on $\mathcal{D}^{\text{train}}$. Among the N candidates of hyper-parameters for model class $\mathcal{M}$, we choose θ_n with the smallest $\overline{MSE}_{\theta_n}$ to be the best-performing parameter, denoted as θ^*. This θ^* is not necessarily the truly best one among all θ in Θ, θ^* is only the best-performing configuration within N samples. However, since we sampled the N candidates uniformly from Θ, performance of the selected θ^* should be close to the truly optimal configuration especially when N is sufficiently large.

While using 5-fold RCV and N sampled θ, we need for fit $5N$ models from class $\mathcal{M}$ in total to determine the optimal configuration θ^* for this model class. Clearly, the larger N is the more likely for us to include the truly optimal configuration in our sampled configurations. Specifically, we choose N to be 500 in this paper.

Since this paper works with two performance metrics: mean-squared-error (MSE) and directional accuracy (DA), there are at least two criterions to identify the optimal model.

MSE-Optimal The first selection criterion identifies the optimal model from a given model class based on models' validation MSEs. Given a class of models, one model is the MSE-optimal of this model class if

- this model achieves at least a DA of 50% on the validation set;
- and, it achieves the lowest MSE on the validation set among all models satisfy the first condition.

If there are more than one models with the same lowest validation MSE and DA greater than 50%, we will choose the one with the best DA to be the optimal model.

DA-Optimal The second criterion selects the best model from candidates based on their validation DAs. Given a class of models, one model is the DA-optimal of this model class if

- this model achieves the highest DA on the validation set.

If there are multiple models with the same lowest accuracy, the model with the lowest validation MSE is chosen to be the DA-optimal.

Let $\mathcal{M}^{\text{MSE}}_{\text{model class}}$ and $\mathcal{M}^{\text{DA}}_{\text{model class}}$ denote the MSE-optimal and DA-optimal models from a particular model class. After identifying optimal models, the performances of $\mathcal{M}^{\text{MSE}}_{\text{model class}}$ and $\mathcal{M}^{\text{DA}}_{\text{model class}}$ on the test set will be proxies of the performance of the specified model class on the prediction task. Note that these optimal models can be identify completely without the test set, therefore, evaluating them on the test set mimics real-world environment: someone builds and trains a model at time t using all information available up to time t, then the model is implemented on company's server and used in production after day t. What traders really care about is the predictive model's performance after day t, called test time performance, since the test time performance is directly related to the profitability of any trading algorithms built on this predictive model. Therefore, one model's performance on the test set is a fair proxy for the business value of this model in real-world.

4.2 Baseline Models: The Naive Predictor and Moving Average Predictors

To better evaluate model performances, we firstly define several dummy models for benchmarking purpose. The simplest model is a naive predictor, $\mathcal{M}_{\text{naive}}$, predicting zero returns all the time. In addition, we define other moving-average predictors denoted as $\mathcal{M}_{\text{MA}(k)}$, where k is a positive integer representing the scope of this model. To predict r_t, $\mathcal{M}_{\text{MA}(k)}$ looks into the past k days and predicts the return to be the average return over the last k trading days.

$$\hat{r}_t = \mathcal{M}_{\text{MA}(k)}(r_0, r_1, r_2, \cdots, r_{t-1}) = \frac{1}{k}\sum_{\tau=t-k}^{t-1} r_t \tag{4.8}$$

As mentioned before, this paper uses all data before December 31, 2018 as the training set and data after January 1, 2019 as the test set. Table 14 shows performances of benchmark models in terms of mean-squared-error (MSE) and directional accuracy (DA).

Table 14: Performances of Benchmark Models

Model	Training MSE	Training DA	Testing MSE	Testing DA
$\mathcal{M}_{\text{naive}}$	4.655	0.716%	4.057	0.538%
$\mathcal{M}_{\text{MA(5)}}$	5.612	50.274%	4.693	50.000%
$\mathcal{M}_{\text{MA(25)}}$	4.811	50.295%	4.248	50.000%
$\mathcal{M}_{\text{MA(50)}}$	4.725	49.536%	4.261	50.000%
$\mathcal{M}_{\text{MA(100)}}$	4.706	49.241%	4.226	44.624%
$\mathcal{M}_{\text{MA(300)}}$	4.676	47.977%	4.060	48.925%

Because returns are rarely exactly zero in the dataset, the directional accuracy of $\mathcal{M}_{\text{naive}}$ model is nearly zero on both training and testing sets. As we expected, those benchmark models never achieve better performances than random guessing (50% accuracy). These results suggest that the crude oil market is efficient (i.e., unpredictable) with respect to

- the information set containing historical returns,
- and naive predictors as well as moving-average predictors.

Then we are going to examine whether other more sophisticated models can attain significantly better test time performances than those above-mentioned benchmark models.

4.3 Linear Models: Autoregressive Integrated Moving Average

One classical model used for time series forecasting is the autoregressive integrated moving average (ARIMA) model.

The autoregressive moving average (ARMA) models the return at time step t, r_t, as a function the series itself and a series of error terms. In particular, an ARMA(p, q) process identifies r_t as a linear combination of p lagged variation of r_t and q noise terms:

$$X_t - \phi_1 X_{t-1} - \cdots - \phi_p X_{t-p} = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} \tag{4.9}$$

More concisely, let L denote the lag operator, an ARMA(p,q) process can be written with the more compact notion

$$\Phi_p(L)r_t = \Theta_q(L)\varepsilon_t \tag{4.10}$$

where $\Phi_p(\cdot)$ and $\Theta_q(\cdot)$ are polynomials up to degree p and q respectively.

One crucial assumption on $\{r_t\}$ is that it has to be stationary. To handle non-stationary processes, one can apply the differencing operator, $1 - L$, iteratively until the differenced series becomes stationary. If the stationarity is achieved after d iterations of differencing, the original series is said to be integrated of order d and can be modelled using an ARIMA(p, d, q) model:

$$\Phi_p(L)(1 - L)^d X_t = \Theta_q(L)\varepsilon_t \tag{4.11}$$

ARIMA models the inter-temporal dependencies naturally, however, ARIMA can only handle linear relationships. In later sections, non-linear models such as support vector machines will be introduced.

This paper searches over 72 configurations of ARIMA models, $\mathcal{M}_{\text{ARIMA}(p,d,q)}$, with different combinations of (p, d, q) specified in table 15. This paper uses the Akaike's information criterion (AIC) of ARIMA models on the training set to identify the optimal configuration. ARIMA(5,0,5), ARIMA(4,0,3) and ARIMA(5,0,4) are the three models attain the three lowest AIC values on training set. Table 16 summarizes the performances of these three optimal models identified.

Table 15: Scope of Orders for ARIMA

Parameter	Scope
p	{0,1,2,3,4,5}
d	{0,1,2}
q	{0,1,2,3,4,5}

Table 16: Performances of Linear Models

Model	Testing MSE	Testing DA
$\mathcal{M}_{\text{ARIMA}(5,0,5)}$	4.074	50.763 %
$\mathcal{M}_{\text{ARIMA}(4,0,3)}$	4.070	51.156 %
$\mathcal{M}_{\text{ARIMA}(5,0,4)}$	4.073	50.567 %

It turns out that ARIMA models perform poorly and merely out-performed benchmark models. Therefore, we may extend our previous conclusion: the crude oil market is efficient with respect to

- the information set containing historical returns,
- and naive predictors, moving-average predictors and ARIMA models.

Since ARIMA models work univariate time series, we can only test the market's efficiency on the information set without news sentiment.

4.4 Support Vector Regression

The previous section shows that the crude oil market is efficient with respect to linear models. In contrast, this section is devoted to non-linear models and answers whether the market is efficient against non-linear models.

Support vector machines (SVM) was firstly proposed by Boser, Guyon and Vapnik as a classification method for hand-written digit recognition (1992). Over the past three decades, SVM has been believed to be the best off-the-shelf classification algorithm.

As mentioned in section 4.1.1, characteristic functions generate 416 predictors in total for each one target r_t, so that the prediction task is in fact a high dimensional problem. SVM models only focus a few training samples termed *support vectors*, therefore, SVMs often produce promising results on high dimensional problems.

Moreover, by using different *kernel functions*, SVMs are capable of transforming these raw features to an even higher dimensional space. For example, if one wishes to classify points in $\mathbb{R}^2$, other algorithms like logistic regressions would classify the point based on $(x_1, x_)$ directly and maps (x_1, x_2) to class labels. Instead, a SVM with polynomial kernel of degree two will classify the point based all combinations of (x_1, x_2) up to degree two, that is, $(x_1, x_2, x_1^2, x_1x_2, x_2^2) \in \mathbb{R}^5$. In this case, the original 2-dimensional input space is transformed into a 5-dimensional feature space by the kernel function. While a SVM is using the Radial Basis Function (RBF) kernel, the original input space is transformed into an infinite-dimensional feature space. This implicit feature engineering enables SVM to explore

more complex patterns in the dataset. Smola and Scholkorf provide a detailed review of training SVMs and theories behind kernel functions in their work (2004).

A few years after the SVM was proposed as a classifier, Drucker and others proposed an extension to original SVM called support vector regression machines (SVR) (Drucker et al. 1997). As the name suggests, SVR is designed for regression problems. It has been shown that SVRs perform reasonably well on high dimensional regression problems by focusing on a few support vectors and engineering features implicitly using kernel functions.

As mentioned in Smola and Scholkorf's work, performances of support vector machines are determined by several hyper-parameters listed in table 17. To choose the optimal configuration of SVR in this paper's prediction task, a RCV algorithm samples 500 configurations from the scope of hyper-parameters in table 17 and evaluates each configuration based on their MSE and DA on the validation set.

Table 17: Scope of Hyper-parameters for Support Vector Regression Machines

Hyper-parameter	Scope
Kernel Type	{Radial Basis Function (RBF) kernel}
γ	$\{10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 10\}$
Tolerance	$\{10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 10\}$
ε	$\{10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 10\}$
C	$\{10^{-10}, 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 10\}$

Table 18 and table 19 presents the optimal models under both criterions and their respective performances.

Table 18: Optimal Hyper-parameters for Support Vector Regression Machines

Model	Kernel	γ	Tolerance	ε	C
$\mathcal{M}_{\text{SVR}}^{\text{MSE}}$	RBF	10^{-6}	1	10^{-3}	10^{-9}
$\mathcal{M}_{\text{SVR}}^{\text{DA}}$	RBF	10^{-7}	10^{-5}	10^{-7}	10

Table 19: Performances of Support Vector Regression Machines

Model	Validation MSE	Validation DA	Testing MSE	Testing DA
$\mathcal{M}_{\text{SVR}}^{\text{MSE}}$	4.655	51.433 %	4.055	54.301 %
$\mathcal{M}_{\text{SVR}}^{\text{DA}}$	4.830	51.665 %	4.399	49.462 %

The MSE-optimal SVR, $\mathcal{M}_{\text{SVR}}^{\text{MSE}}$, out-performs the naive predictor and all linear models in terms of both test-time MSE and DA. In contrast, $\mathcal{M}_{\text{DA}}^{\text{MSE}}$ only attains unsatisfactory results compared with our benchmarks.

Given $\mathcal{M}_{\text{SVR}}^{\text{MSE}}$'s superior performance, we can conclude the crude oil market is inefficient with respect to

- (information set) the information set consists of both historical returns and news sentiments,
- (predictive models) support vector regression machines;
- (searching technology) randomized cross validation and select the optimal model based on validation MSE.

4.5 Random Forests

Another class of non-linear methods used widely is the random forest. Breiman proposed an ensemble model based on traditional tree methods called random forest (2001). A forest as an ensemble of independently trained trees reduces the variance of prediction and achieves a better performance compared with one single tree predictor. Let p denote the number of independent variables for prediction ($p = 416$ here). Training each tree in the forest using all p features can cause over-fitting problems and lead to poor test time performance. Therefore, each independent tree predictor in the forest is trained only using a random subset of p features, which constitutes the randomness of a random forest. This randomness on feature selection helps random forests to generalize better and achieves even lower loss on the test set. Typically, each tree in the forest is only trained on $\log_2(p)$ features.

Table 20 enumerates key hyper-parameters of a random forest predictor and corresponding scopes our cross validation procedure searches over.

Table 20: Scope of Hyper-parameters for Random Forests

Hyper-parameter	Scope
n Number of trees	$\{1, 2, 3, \cdots, 200\}$
f Max num. of features for each tree	$\{p, \log_2(p)\}$
d Max depth of each tree	$\{10, 14, 19, 24, \cdots, 100, 105, 110, \infty\}$
m_1 Min amount of samples required to split an internal node	$\{2, 5, 10\}$
m_2 Minimum number of samples required at each leaf node	$\{1, 2, 4\}$
What dataset is used to construct each tree	{bootstrapped samples, entire training dataset}

Table 20 summarizes configurations of $\mathcal{M}_{\text{RF}}^{\text{MSE}}$ and $\mathcal{M}_{\text{RF}}^{\text{DA}}$ identified.

Table 21: Optimal Hyper-parameters for Random Forests

Model	n	f	d	m_1	m_2	Training Samples
$\mathcal{M}_{\text{RF}}^{\text{MSE}}$	96	$\log_2(p)$	10	10	2	Bootstrapped Samples
$\mathcal{M}_{\text{RF}}^{\text{DA}}$	41	p	14	5	4	Entire Training Set

After identifying optimal models, these models are evaluated using the test set, and table 21 reports the performances. Since we search over a sufficiently large scope of hyper-parameters for random forests, those test-time performances in table 21 should be close to the best a random forest can do on this return prediction problem. It turns out that the performances of random forests only achieve an unsatisfactory result in terms of testing MSE. Both optimal models perform worse than a naive predictor, which attains a testing MSE of 4.057.

Table 22: Performances of Random Forests

Model	Validation MSE	Validation DA	Testing MSE	Testing DA
$\mathcal{M}_{\text{RF}}^{\text{MSE}}$	4.675	50.464 %	4.148	48.387 %
$\mathcal{M}_{\text{RF}}^{\text{DA}}$	5.716	51.960 %	4.753	53.226 %

In terms of the directional accuracy, $\mathcal{M}_{\text{RF}}^{\text{DA}}$ achieves better results than random guessing on both validation and test sets. However, a market is predictable only if a model can consistently achieve better-than-guessing results. The statistics in table 22 are not sufficient

to conclude the market is predictable. Therefore, we conclude the crude oil market to be efficient with respect to

- an information set containing both historical returns and news sentiments,
- and random forest models.

4.6 Long Short-Term Memory Recurrent Neural Networks

An ARIMA model captures the intertemporal correlation among independent variables explicitly. However, ARIMA models are not capable of modelling complex non-linearities. The superior performance of SVRs indicates that considering non-linear interactions among independent variables can improve model performance significantly.

Unfortunately, even though SVRs and random forests are capable to model complex, they squeeze all independent variables and disregard the orders among independent variables. Hence, SVRs and random forests are not able to utilize information from the sequential structures (orders) of independent variables.

In current literature, neural networks have been used widely to capture non-linearity among independent variables. One special type of neural works termed recurrent neural networks (RNN) is designed to model both non-linearities and inter-temporal correlations. However, conventional RNNs suffer from vanishing and exploding gradient problems and becomes impotent on longer time series. In section 2.2, the ACF and PACF plots have identified possible seasonality in crude oil returns. Modelling seasonality requires the RNN to pay attention to inter-temporal dependencies over a longer period of time.

Hochreiter and Schmidhuber proposed the long short-term memory (LSTM) cell for RNNs, this novel architecture allows RNNs to focus on inter-temporal dependencies over both short and long periods (1997). All RNNs trained and evaluated in this paper are based on this LSTM architecture.

Table 23 summarizes the scope of hyper-parameters for LSTM RNNs.

Table 23: Scope of Hyper-parameters for LSTM RNNS

Hyper-parameter	Scope
Epochs of training	$\{5, 6, 7, 8, \cdots, 18, 19, 20, 25, 30, 35, \cdots, 200\}$
h Size of RNN hidden layer	$\{32, 64, 128, 256, 512, 1024\}$
ℓ Number of RNN hidden layers	$\{1, 2, 3\}$
p_{rnn} Dropout probability in RNN hidden layers	$\{0, 0.25, 0.5\}$
p_{fc} Dropout probability in the output layer	$\{0, 0.25, 0.5\}$
B Batch size	$\{32, 128, 512\}$
α Learning rate	$\{10^{-5}, 3 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 0.01, 0.03, 0.1, 0.3\}$

Table 24 and table 25 present two optimal models and their test time performances. Because LSTM RNNs are capable to capture all of non-linearities, inter-temporal dependencies over short time periods (inter-day transitions) and inter-temporal dependencies over longer time periods (seasonalities), both $\mathcal{M}^{\text{MSE}}_{\text{LSTM}}$ and $\mathcal{M}^{\text{DA}}_{\text{LSTM}}$ out-perform other models examined before.

Table 24: Optimal Hyper-parameters for LSTM RNNs

Model	Epochs	h	ℓ	p_{rnn}	p_{fc}	B	α
$\mathcal{M}^{\text{MSE}}_{\text{LSTM}}$	40	32	2	0.0	0.5	512	10^{-5}
$\mathcal{M}^{\text{DA}}_{\text{LSTM}}$	12	128	2	0.0	0.25	512	10^{-5}

Table 25: Performances of LSTM RNNs

Model	Validation MSE	Validation DA	Testing MSE	Testing DA
$\mathcal{M}^{\text{MSE}}_{\text{LSTM}}$	4.192	51.609 %	4.043	54.012 %
$\mathcal{M}^{\text{DA}}_{\text{LSTM}}$	4.888	54.480 %	4.041	54.011 %

Statistics in table 25 suggest that LSTM RNNs consistently attain non-trivial performances on both validation and test sets. Therefore, crude oil returns are in fact predictable using LSTM RNNs and the market is concluded to be inefficient with respect to

- An information set consists of both historical returns and news sentiment,
- LSTM RNN models,
- and model selection techniques based on either MSE or DA on the validation set.

4.7 Taking the Day-of-the-Week Effect into Consideration

In section 2.3, we have shown that crude oil returns experience significant day-of-the-week effect. In particular, Mondays are more likely to experience negative returns compared with other days. Therefore, it is reasonable to conjecture that the underlying dynamics of returns on Mondays might be different from the dynamics of returns of other days. To utilize this fact, we select and train two separate models for returns on Mondays and returns on the rest of the week.

The same procedure identifies the optimal configuration of each model class for two separate datasets:

- a dataset with Mondays only consisting of 889 training samples and 34 testing samples,
- and another dataset excluding all Mondays consisting of 3,857 training samples and 152 tests samples.

Optimal models identified for both datasets are reported in table 26, table 27 and table 28. The notation for optimal models is similar, for example, $\mathcal{M}^{\text{MSE}}_{\text{RF, Mondays}}$ represents the MSE-optimal random forest for predicting returns on Mondays and $\mathcal{M}^{\text{MSE}}_{\text{SVR, Other Days}}$ indicates the MSE-optimal SVR for predicting returns on Tuesdays, Wednesdays, Thursdays and Fridays.

Table 26: Optimal Hyper-parameters for Random Forests on Restricted Datasets

Model	n	f	d	m_1	m_2	Training Samples
$\mathcal{M}^{\text{MSE}}_{\text{RF, Mondays}}$	115	$\log_2(p)$	38	10	4	Bootstrapped Samples
$\mathcal{M}^{\text{DA}}_{\text{RF, Mondays}}$	116	$\log_2(p)$	38	2	1	Entire Training Set
$\mathcal{M}^{\text{MSE}}_{\text{RF, Other Days}}$	88	$\log_2(p)$	10	5	4	Bootstrapped Samples
$\mathcal{M}^{\text{DA}}_{\text{RF, Other Days}}$	157	p	10	2	1	Entire Training Set

Table 27: Optimal Hyper-parameters for Support Vector Regressions on Restricted Datasets

Model	Kernel	γ	Tolerance	ε	C
$\mathcal{M}^{\text{MSE}}_{\text{SVR, Mondays}}$	RBF	10^{-10}	10^{-3}	10^{-4}	10
$\mathcal{M}^{\text{DA}}_{\text{SVR, Mondays}}$	RBF	10^{-6}	0.1	10^{-6}	1
$\mathcal{M}^{\text{MSE}}_{\text{SVR, Other Days}}$	RBF	0.1	0.1	10^{-4}	10
$\mathcal{M}^{\text{DA}}_{\text{SVR, Other Days}}$	RBF	10^{-9}	0.1	10^{-7}	1

Table 28: Optimal Hyper-parameters for LSTM RNNs on Restricted Dataset

Model	Epochs	h	ℓ	p_{rnn}	p_{fc}	B	α
$\mathcal{M}^{\text{MSE}}_{\text{LSTM, Mondays}}$	45	128	2	0.5	0.0	128	10^{-4}
$\mathcal{M}^{\text{DA}}_{\text{LSTM, Mondays}}$	75	1024	3	0.0	0.25	128	0.01
$\mathcal{M}^{\text{MSE}}_{\text{LSTM, Other Days}}$	14	512	3	0.0	0.5	32	0.001
$\mathcal{M}^{\text{DA}}_{\text{LSTM, Other Days}}$	12	512	3	0.0	0.0	32	0.001

Performances of models are reported in table 29, the best performances are in bold font. It turns out that optimal models for both datasets are LSTM RNNs. In particular, $\mathcal{M}^{\text{MSE}}_{\text{LSTM, Mondays}}$ achieves a superior performance in terms of the directional accuracy in the test set. However, all three types of models used are data-intensive but there are only 889 training samples of Mondays. Therefore, none of them delivers a significantly better result compared with models previously trained using the entire dataset.

Table 29: Performances of Models on Restricted Datasets

Model	Validation MSE	Validation DA	Testing MSE	Testing DA
$\mathcal{M}^{\text{MSE}}_{\text{SVR, Mondays}}$	0.659	52.984 %	0.943	41.176 %
$\mathcal{M}^{\text{DA}}_{\text{SVR, Mondays}}$	0.681	54.887 %	1.042	44.118 %
$\mathcal{M}^{\text{MSE}}_{\text{RF, Mondays}}$	0.655	55.574 %	0.919	47.059 %
$\mathcal{M}^{\text{DA}}_{\text{RF, Mondays}}$	0.672	56.461 %	**0.939**	55.882 %
$\mathcal{M}^{\text{MSE}}_{\text{LSTM, Mondays}}$	**0.484**	51.479 %	0.971	**57.143** %
$\mathcal{M}^{\text{DA}}_{\text{LSTM, Mondays}}$	0.673	**56.819** %	1.080	42.857 %
$\mathcal{M}^{\text{MSE}}_{\text{SVR, Other Days}}$	5.575	52.605 %	4.762	53.289 %
$\mathcal{M}^{\text{DA}}_{\text{SVR, Other Days}}$	5.583	52.631 %	4.776	53.289 %
$\mathcal{M}^{\text{MSE}}_{\text{RF, Other Days}}$	5.606	50.997 %	4.799	46.711 %
$\mathcal{M}^{\text{DA}}_{\text{RF, Other Days}}$	6.844	52.449 %	5.499	51.974 %
$\mathcal{M}^{\text{MSE}}_{\text{LSTM, Other Days}}$	**4.947**	51.255 %	**4.749**	**53.290** %
$\mathcal{M}^{\text{DA}}_{\text{LSTM, Other Days}}$	5.761	**54.836** %	4.760	**53.290** %