MATH 304 Numerical Analysis and Optimization Project

Traffic flow prediction by using LSR and SVR

Xuchen Gong

(xg54)

Task

Predict Traffic Flow

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	TMU ID	Legacy TM	Site Name										
2	233C060F!	30016383	TMU Site 9956/1 on link A1081 westbound between A505/B653 and M1 J10; GPS Ref: 509008;218723; Westbound										
3													
4	Local Date	Local Tim	Day Type	Total Carr	Total Flow	Total Flow	Total Flow	Total Flow	Speed Val	Quality Inc	Network L	NTIS Model Version	
5	2016/6/1	0:14:00	7	151	145	2	0	4	80.71	15	2E+08	4	
6	2016/6/1	0:29:00	7	131	116	6	4	5	81.01	15	2E+08	4	
7	2016/6/1	0:44:00	7	147	140	1	1	5	81.6	15	2E+08	4	
8	2016/6/1	0:59:00	7	101	94	3	2	2	81.07	15	2E+08	4	
9	2016/6/2	1:14:00	9	194	184	3	5	2	79.08	15	2E+08	4	
10	2016/6/2	1:29:00	9	129	114	7	4	4	80.51	15	2E+08	4	
11	2016/6/2	1:44:00	9	83	74	2	3	4	80.4	15	2E+08	4	
12	2016/6/2	1:59:00	9	74	60	4	5	5	83.23	15	2E+08	4	
13	2016/6/2	2:14:00	9	58	39	7	5	7	82.68	15	2E+08	4	
14	2016/6/2	2:29:00	9	36	28	3	2	3	79.13	15	2E+08	4	
15	2016/6/2	2:44:00	9	68	61	4	1	2	83.22	15	2E+08	4	
16	2016/6/2	2:59:00	9	72	59	6	4	3	83.48	15	2E+08	4	
17	2016/6/2	3:14:00	9	70	51	6	3	10	82.95	15	2E+08	4	
18	2016/6/2	3:29:00	9	68	62	3	1	2	80.85	15	2E+08	4	
19	2016/6/2	3:44:00	9	61	52	4	2	3	79.16	15	2E+08	4	
20	2016/6/2	3:59:00	9	75	64	4	4	3	80.47	15	2E+08	4	
21	2016/6/2	4:14:00	9	90	73	9	3	5	82.18	15	2E+08	4	
22	2016/6/2	4:29:00	9	95	79	7	2	7	81.16	15	2E+08	4	
23	2016/6/2	4:44:00	9	128	109	13	2	4	79.87	15	2E+08	4	
24	2016/6/2	4:59:00	9	161	137	11	7	6	80.75	15	2E+08	4	
25	2016/6/2	5:14:00	9	227	200	13	7	7	78.43	15	2E+08	4	
26	2016/6/2	5:29:00	9	290	261	16	7	6	78.35	15	2E+08	4	

X: Local Time
Y: Total Carriage Flow

Function *hours()*
Convert X to double
1.23
…
…
23.68

Evaluation Metrics

$$mse = \sum_{i=1}^{N} \frac{1}{N} (\widehat{y}_i - y_i)^2$$

$$r^2 = \frac{[N \sum_{i=1}^{N}(\widehat{y}_i y_i) - (\sum_{i=1}^{N} \widehat{y}_i)(\sum_{i=1}^{N} y_i)]^2}{[N \sum_{i=1}^{N} \widehat{y}_i{}^2 - (\sum_{i=1}^{N} \widehat{y}_i)^2][N \sum_{i=1}^{N} y_i{}^2 - (\sum_{i=1}^{N} y_i)^2]}$$

$$r^2 = \frac{S_t - S_r}{S_t}$$

$\boldsymbol{S_t}$: Total sum of the squares around the mean for the dependent variable.

$\boldsymbol{S_r}$: Sum of the squares of residuals around the regression line.

$\boldsymbol{S_t - S_r}$: Quantifies the improvement due to describing data in terms of a straight line rather than as an average value.

Perfect fit
$S_r = 0$ and $r^2 = 1$
The line explains 100 percent of the variability of the data.

Algorithms

Least-square Regression (LSR) and Support Vector Regression (SVR) are applied in this project to predict the traffic flow at different times of a day.

LSR (Polynomial)

$$f(t) = a_n t^n + a_{n-1} t^{n-1} + \dots + a_2 t^2 + a_1 t^1 + a_0$$
$$where\ n = 1, 2, \dots, 9\dots$$

SVR (Polynomial)

$$k(x, z) = (x^T z)^d$$

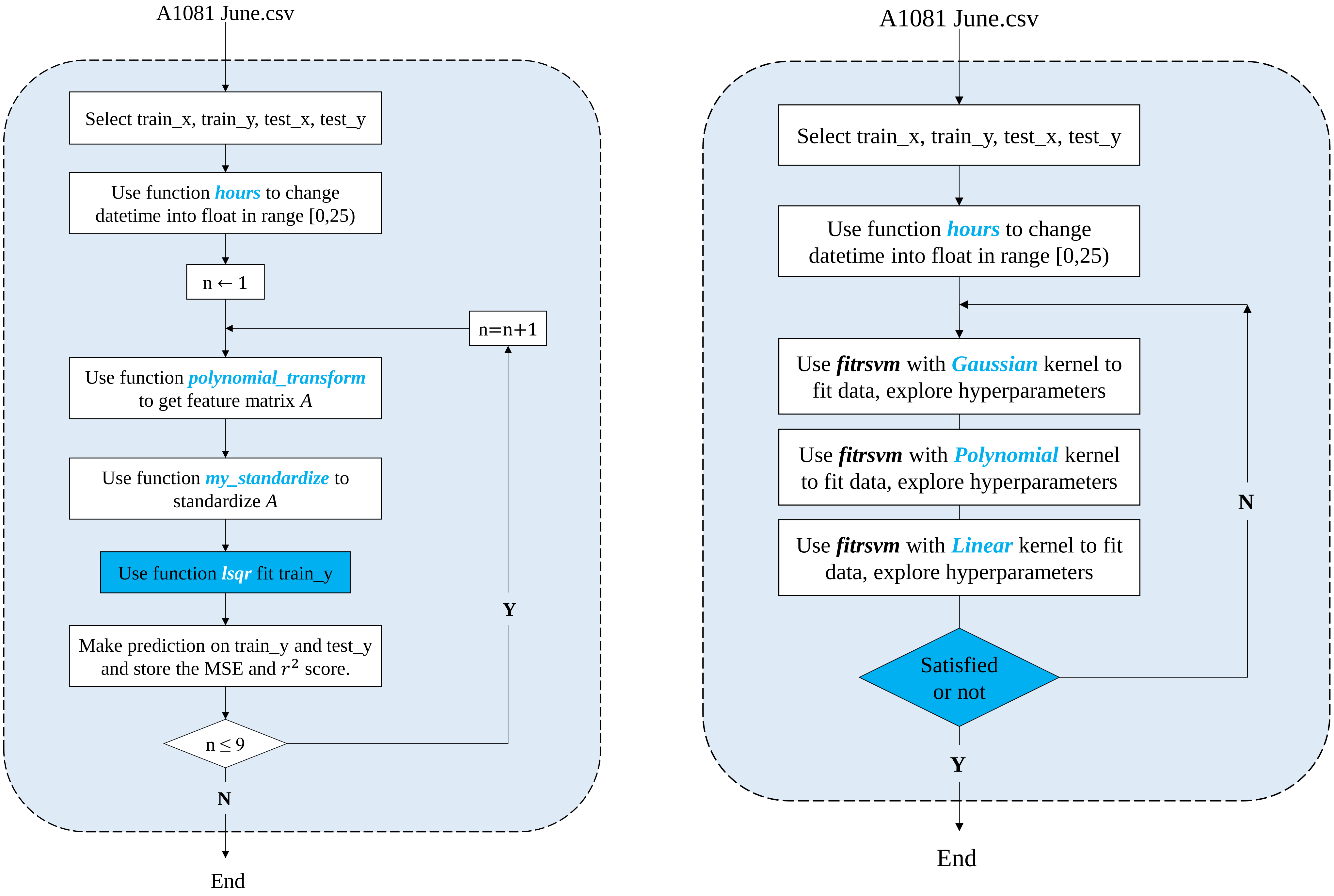
SVR (Linear)

$$k(x, z) = x^T z$$

SVR (Gaussian)

$$k(x, z) = exp(-\frac{\|x - z\|_2^2}{2\sigma^2})$$

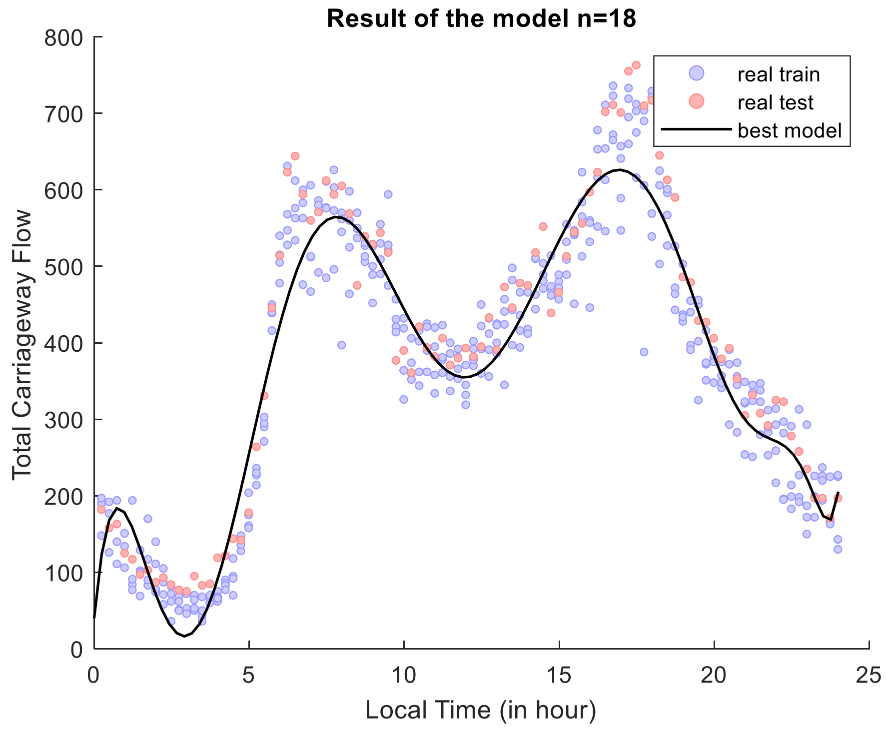
Flow chart


A1081 June.csv
Select train_x, train_y, test_x, test_y
Use function hours to change datetime into float in range [0,25)
n ← 1
n=n+1
Use function polynomial_transform to get feature matrix A
Use function my_standardize to standardize A
Use function lsqr fit train_y
Make prediction on train_y and test_y and store the MSE and r^2 score.
n ≤ 9
Y
N
End
A1081 June.csv
Select train_x, train_y, test_x, test_y
Use function hours to change datetime into float in range [0,25)
Use fitrsvm with Gaussian kernel to fit data, explore hyperparameters
Use fitrsvm with Polynomial kernel to fit data, explore hyperparameters
Use fitrsvm with Linear kernel to fit data, explore hyperparameters
Satisfied or not
N
Y
End

Experimental Results

Polynomial Regression

Model	Training error		Test error	
	MSE	r^2	MSE	r^2
n = 1	29834	0.10841	32158	0.11421
n = 2	13194	0.60569	15084	0.58450
n = 3	12732	0.61952	14313	0.60576
n = 4	12687	0.62084	14152	0.61019
n = 5	11969	0.64230	13453	0.62945
n = 6	6315.0	0.81128	6852.6	0.81125
n = 7	6269.7	0.81263	6816.2	0.81235
n = 8	3080.2	0.90794	3346.1	0.90783
n = 9	2953.2	0.91174	3329.7	0.90829
n = 10	2891.8	0.91358	3295.7	0.90922
n = 11	2869.1	0.91426	3284.6	0.90953
n = 12	2870.4	0.91422	3284.6	0.90953
n = 13	2870.4	0.91422	3289.6	0.90939
n = 14	2878.8	0.91397	3309.6	0.90884
n = 15	2844.0	0.91500	3188.3	0.91218
n = 16	2826.1	0.91554	3135.0	0.91365
n = 17	2814.0	0.91591	3094.0	0.91478
n = 18	2813.1	0.91593	3075.3	0.91529
n = 19	2825.6	0.91556	3082.2	0.91510
n = 20	2851.3	0.91479	3113.6	0.91424

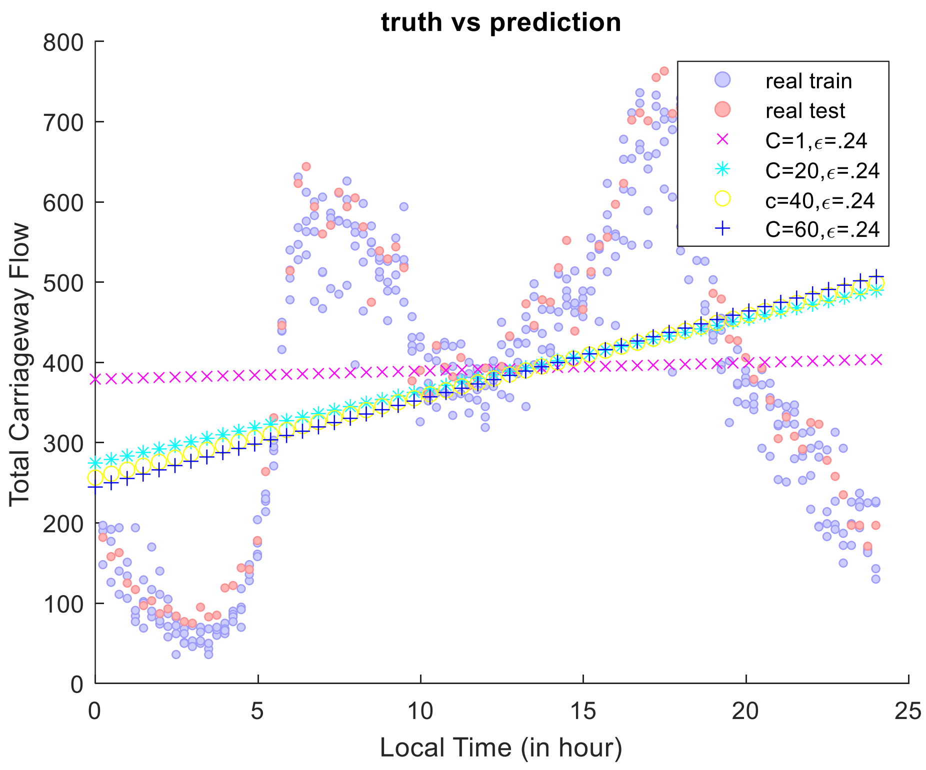


The actual flow and the flow predicted by the polynomial model n=18.

SVR (Linear)

Model	SVR (Linear kernel)			
Setting	Default	Case1	Case2	Optimized
Train MSE	30325.68	30021.82	30019.77	30019.77
Train r^2	0.094	0.1028	0.1029	0.1029
Test MSE	32012.13	31699.00	31696.34	31696.34
Test r^2	0.118	0.1268	0.1269	0.1269

Opt. hp	BoxConstraint=58, KernelScale=13.2, Epsilon=0.236
Case 1	BoxConstraint=20 , KernelScale=13.2, Epsilon=0.236
Case 2	BoxConstraint=19, KernelScale=12, Epsilon=0.01
Optimized	BoxConstraint=19, KernelScale=12, Epsilon=0.01

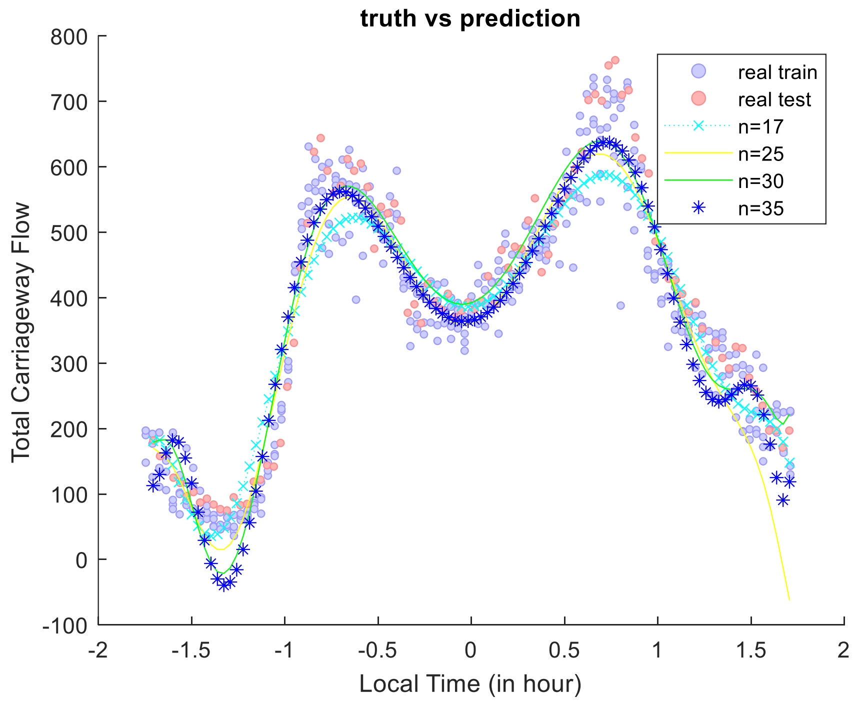


Explore effects of BoxConstraint on SVR(Linear).

SVR (Polynomial)

Model	SVR (Polynomial kernel)			
Setting	Default	Case1	Case2	Optimized
Train MSE	13963.17	3399.41	2701.63	2701.63
Train r^2	0.582713	0.8984	0.9193	0.9193
Test MSE	16819.28	4029.85	2974.85	2974.85
Test r^2	0.5367	0.8890	0.9181	0.9181

Opt. hp	BoxConstraint=521.78, KernelScale=2.47, Epsilon=0.22
Case 1	PolynomialOrder=17, Opt. hp
Case 2	PolynomialOrder=25, Opt. hp
Optimized	PolynomialOrder=25, Opt. hp

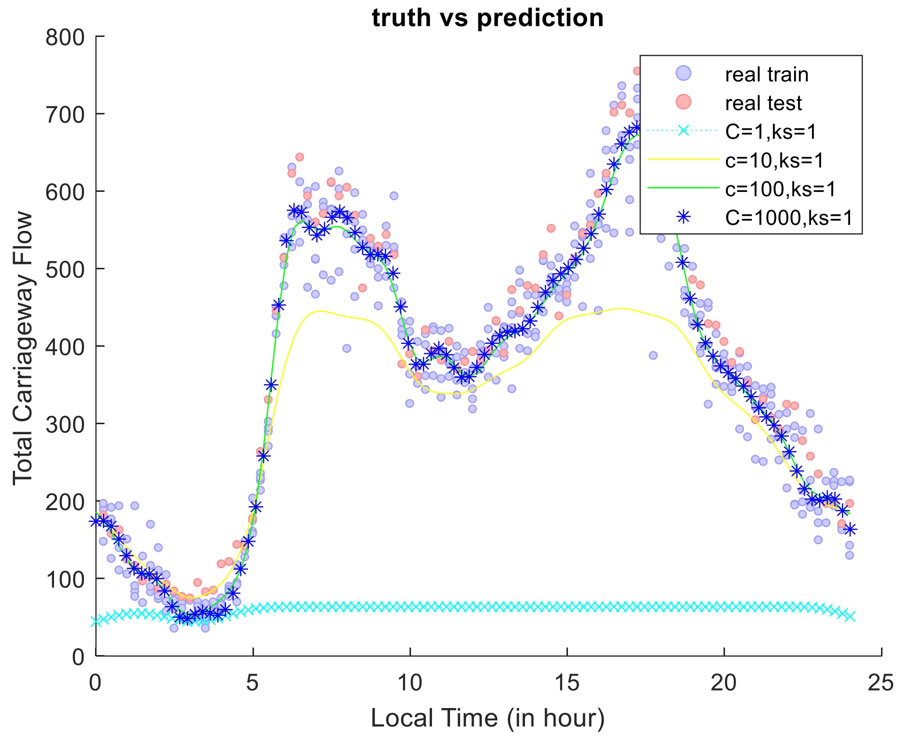


Explore effects of PolynomialOrder on SVR (Polynomial).

SVR (Gaussian)

Model	SVR (Gaussian kernel)			
Setting	Default	Case1	Case2	Optimized
Train MSE	1587.85	1594.13	1752.91	1752.91
Train r^2	0.9525	0.9506	0.9476	0.9476
Test MSE	1503.10	1591.13	1470.62	1470.62
Test r^2	0.9586	0.9562	0.9595	0.9595

Opt. hp	BoxConstraint=137, KernelScale=1, Epsilon=0.45
Case 1	BoxConstraint=150, KernelScale=1, Epsilon=0.45
Case 2	BoxConstraint=185, KernelScale=1.3, Epsilon=0.45
Optimized	BoxConstraint=185, KernelScale=1.3, Epsilon=0.45

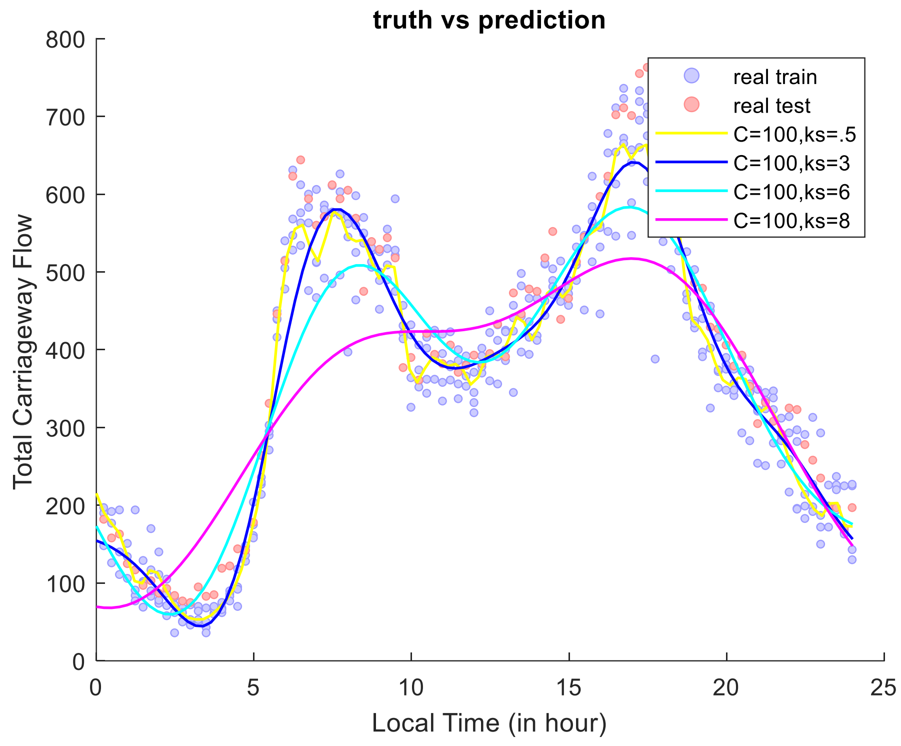


Explore effects of BoxConstraints on SVR (Gaussian).

SVR (Gaussian)

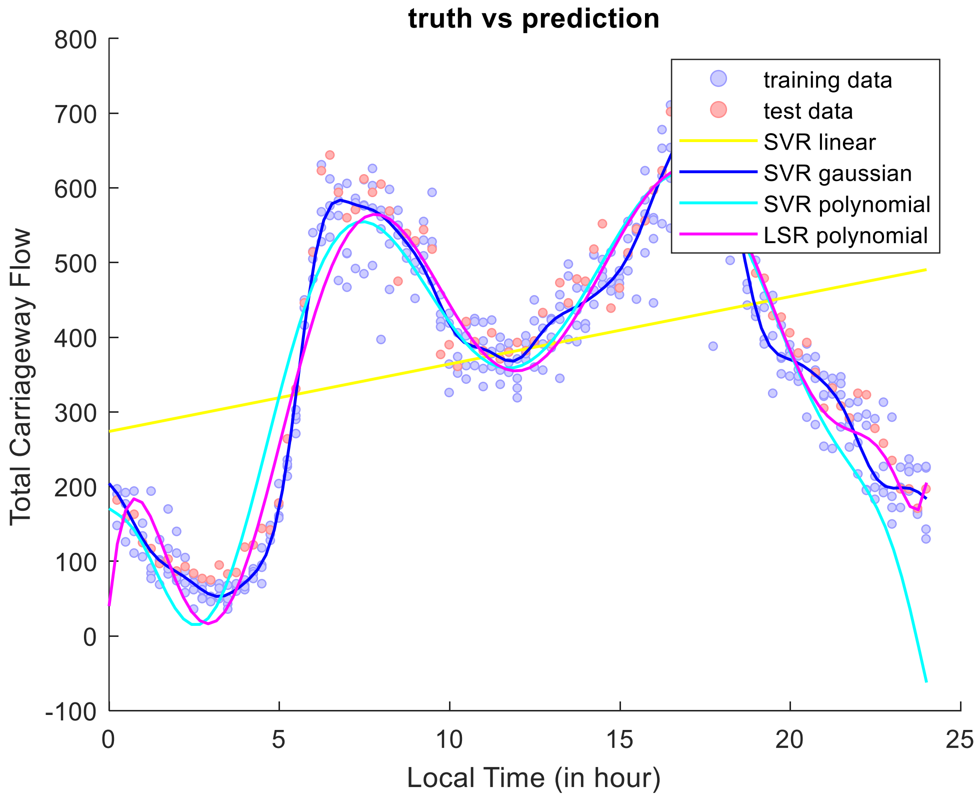
Model	SVR (Gaussian kernel)			
Setting	Default	Case1	Case2	Optimized
Train MSE	1587.85	1594.13	1752.91	1752.91
Train r^2	0.9525	0.9506	0.9476	0.9476
Test MSE	1503.10	1591.13	1470.62	1470.62
Test r^2	0.9586	0.9562	0.9595	0.9595

Opt. hp	BoxConstraint=137, KernelScale=1, Epsilon=0.45
Case 1	BoxConstraint=150, KernelScale=1, Epsilon=0.45
Case 2	BoxConstraint=185, KernelScale=1.3, Epsilon=0.45
Optimized	BoxConstraint=185, KernelScale=1.3, Epsilon=0.45



Explore effects of KernelScale on SVR (Gaussian).

Compare performance



Predictions made by the optimized models.

Model	LSR polynomial	SVR linear	SVR polynomial	SVR gaussian
Train MSE	2813.1	30019.77	2701.63	1752.91
Train r^2	0.91593	0.1029	0.9193	0.9476
Test MSE	3075.3	31696.34	2974.85	1470.62
Test r^2	0.91529	0.1269	0.9181	0.9595

Discussion

Factors that impact performance

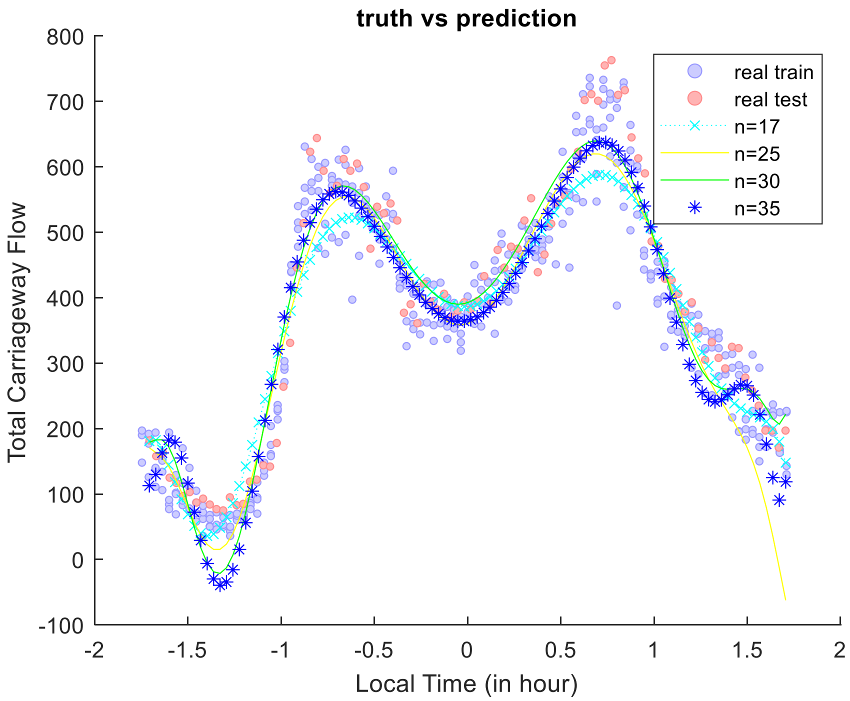
PolynomialOrder

Model	SVR (Polynomial kernel)			
Setting	Default	Case1	Case2	Optimized
Train MSE	13963.17	3399.41	2701.63	2701.63
Train r^2	0.582713	0.8984	0.9193	0.9193
Test MSE	16819.28	4029.85	2974.85	2974.85
Test r^2	0.5367	0.8890	0.9181	0.9181

Model	Training error		Test error	
	MSE	r^2	MSE	r^2
n = 1	29834	0.10841	32158	0.11421
n = 2	13194	0.60569	15084	0.58450
n = 3	12732	0.61952	14313	0.60576
n = 4	12687	0.62084	14152	0.61019
n = 5	11969	0.64230	13453	0.62945
n = 6	6315.0	0.81128	6852.6	0.81125
n = 7	6269.7	0.81263	6816.2	0.81235
n = 8	3080.2	0.90794	3346.1	0.90783
n = 9	2953.2	0.91174	3329.7	0.90829
n = 10	2891.8	0.91358	3295.7	0.90922
n = 11	2869.1	0.91426	3284.6	0.90953
n = 12	2870.4	0.91422	3284.6	0.90953
n = 13	2870.4	0.91422	3289.6	0.90939
n = 14	2878.8	0.91397	3309.6	0.90884
n = 15	2844.0	0.91500	3188.3	0.91218
n = 16	2826.1	0.91554	3135.0	0.91365
n = 17	2814.0	0.91591	3094.0	0.91478
n = 18	2813.1	0.91593	3075.3	0.91529
n = 19	2825.6	0.91556	3082.2	0.91510
n = 20	2851.3	0.91479	3113.6	0.91424

Factors that impact performance

PolynomialOrder



When polynomial order n is relatively small, the larger the n is, the better the performance.

While a larger n also means a bigger risk of overfitting.

Explore effects of PolynomialOrder on SVR (Polynomial).

Factors that impact performance

BoxConstraint

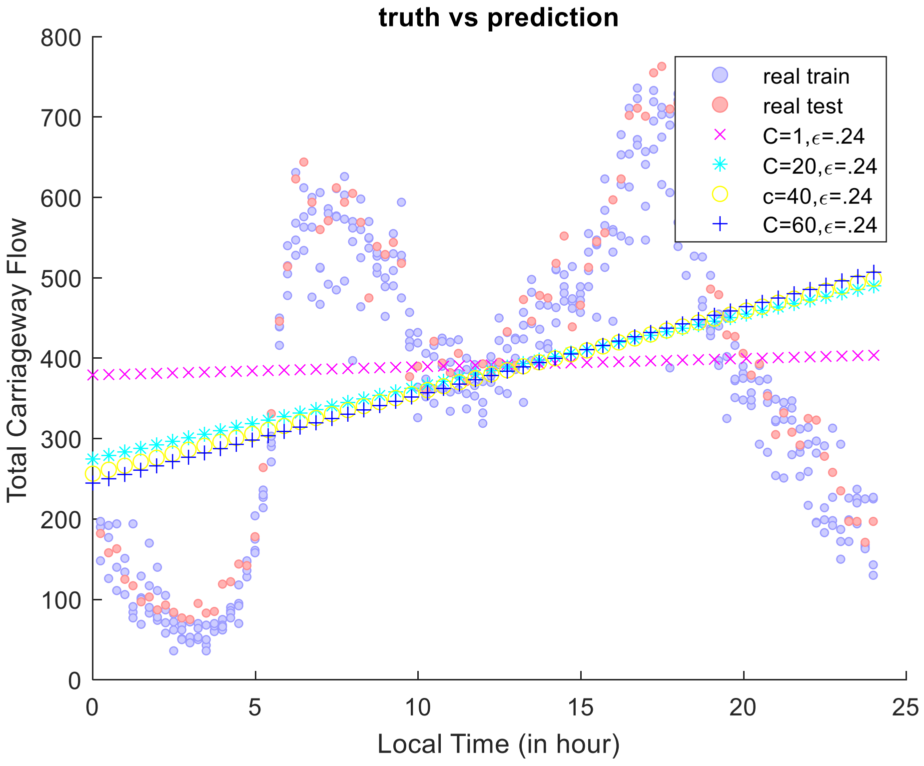
$$
\begin{aligned}
&minimize \quad \frac{1}{2}\|W\|_2^2 + C\sum_{i=1}^{N}(\xi_i - \xi_i{}^*) \\
&subject \quad y_i - (Wx_i + b) \leq \epsilon + \xi_i{}^*, \qquad \xi_i{}^* \geq 0 \\
&\qquad\quad Wx_i + b - y_i \leq \epsilon + \xi_i{}^*, \qquad \xi_i{}^* \geq 0
\end{aligned}
$$

Large C: lower bias, higher variance
Small C: higher bias, lower variance

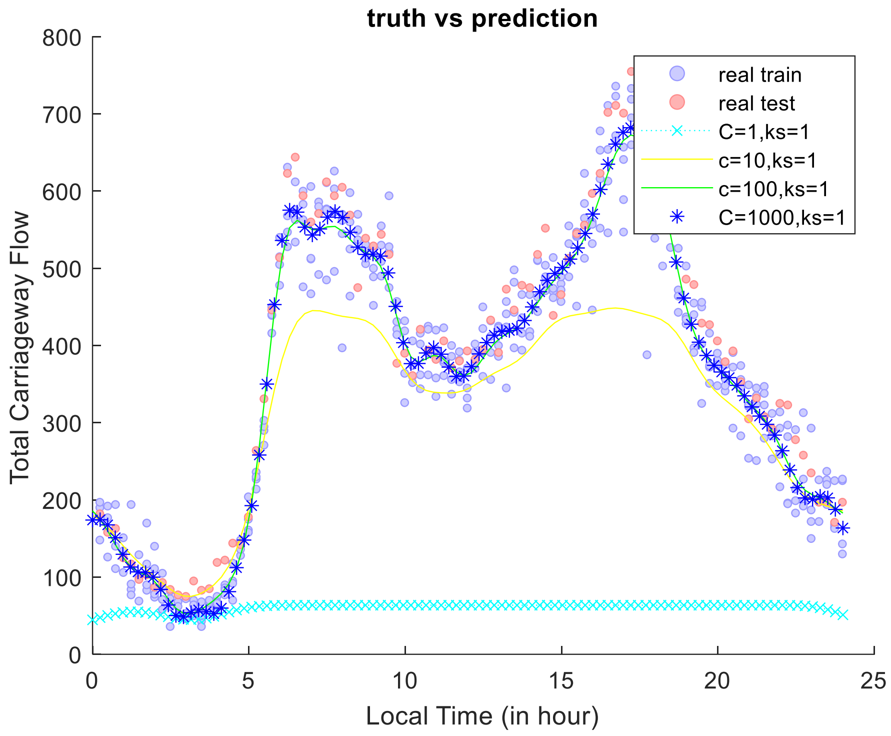
C determines the amount up to which deviations larger than ϵ are tolerated.
A small C helps prevent overfitting (regularization).

Factors that impact performance

BoxConstraint



Explore effects of BoxConstraint on SVR(Linear).



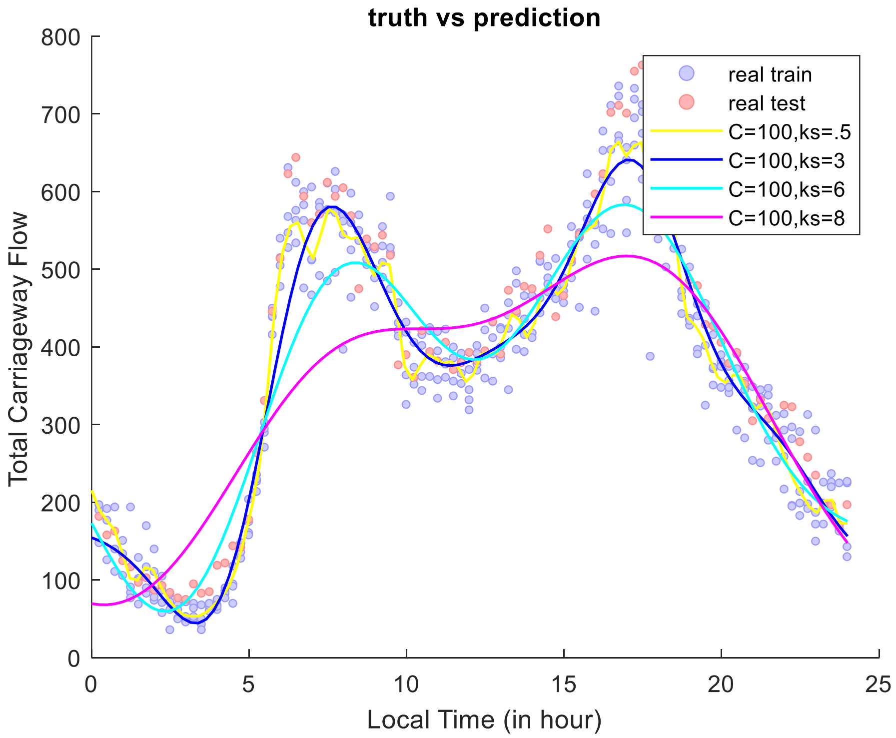
Explore effects of BoxConstraints on SVR (Gaussian).

Factors that impact performance

KernelScale

The software divides all elements of the predictor matrix X by the value of KernelScale. KernelScale determines the extent to which the features vary smoothly.

Large KernelScale: higher bias, lower variance
Small KernelScale: lower bias, higher variance



Explore effects of KernelScale on SVR (Gaussian).

Limits of my approach

1. Other hyperparameters of SVR (Polynomial) also plays an important role and are under-explored in our project currently.

2. For SVR (Polynomial) and SVR (Gaussian), we directly use the suggested value in in our trials instead.

3. Control variable, not a best method.

Improvements

1. Given more time, we plan to first decide on the bond of each hyperparameter and then do gridsearch.

2. Do random search for a large number of times.

Conclusion

Least-square Regression (LSR) and Support Vector Regression (SVR) are applied in this project to predict the traffic flow at different times of a day.

LSR (Polynomial)

SVR (Linear)

SVR (Polynomial)

SVR (Gaussian)

PolynomialOrder

When polynomial order n is relatively small, the larger the n is, the better the performance.

While a larger n also means a bigger risk of overfitting.

BoxConstraint

Large C: lower bias, higher variance
Small C: higher bias, lower variance

KernelScale

Large KernelScale: higher bias, lower variance
Small KernelScale: lower bias, higher variance

Thank you