MATH304 Numerical Analysis and Optimization

Project – Traffic flow prediction using LSR and SVR

Guyun Lu
guyunlu@dukekunshan.edu

***Abstract*—This project explores traffic flow prediction using Least Squares Regression (LSR) and Support Vector Regression (SVR) models. We analyze traffic data from Highways England, focusing on Thursdays in June 2016. The study compares the performance of LSR with different polynomial degrees and SVR with linear, Gaussian, and polynomial kernels. We optimize hyperparameters and evaluate model performance using Mean Squared Error (MSE) and R-squared (R^2) metrics. Our results demonstrate the superiority of the Gaussian kernel SVR in capturing complex traffic patterns, offering insights into effective traffic flow prediction techniques.**

***Index Terms*—Traffic flow prediction, Support Vector Regression (SVR), Least Squares Regression (LSR), Kernel functions, Hyperparameter optimization, Time series analysis**

I. OVERVIEW

In this project, we developed and compared various models to predict traffic flow based on historical data. Our approach involved the following key steps:

1) **Data Preprocessing:** We utilized traffic flow data from Highways England, specifically focusing on Thursdays in June 2016. The data was preprocessed by converting timestamps to minutes and splitting into training (first four Thursdays) and testing (last Thursday) sets.
2) **Least Squares Regression (LSR):** We implemented LSR models with polynomial degrees ranging from 1 to 9. Both analytic solutions and MATLAB's `fit` function were employed to ensure consistency in results.
3) **Support Vector Regression (SVR):** We explored SVR models with three different kernels:
   - Linear kernel
   - Gaussian (RBF) kernel
   - Polynomial kernel

   For each kernel, we tested four configurations: default parameters, two manually set parameter combinations, and one optimized configuration.
4) **Hyperparameter Optimization:** We utilized MATLAB's optimization tools to fine-tune SVR model parameters, aiming to enhance prediction accuracy.
5) **Comparative Analysis:** We compared the performance of LSR and SVR models, identifying the most effective approach for traffic flow prediction in our dataset, using Mean Squared Error (MSE) and R-squared (R^2) metrics for both training and test datasets..

This comprehensive approach allowed us to gain insights into the strengths and limitations of different regression techniques in the context of traffic flow prediction, providing a foundation for future improvements in this critical area of urban planning and traffic management.

II. MATHEMATICAL FORMULATION AND IMPLEMENTATION

A. General Process

In this study, we predict traffic flow using Least Squares Regression (LSR) and Support Vector Regression (SVR). The data analyzed includes traffic flow on Thursdays during June 2016, collected by Highways England. [1]

Fig. 1. outlines the general process. Initially, the original data is pre-processed. The first four Thursdays are used as training data, while the last Thursday is reserved for testing. Additionally, all timestamps (hour, minute, second) are converted to minutes for ease of modeling.

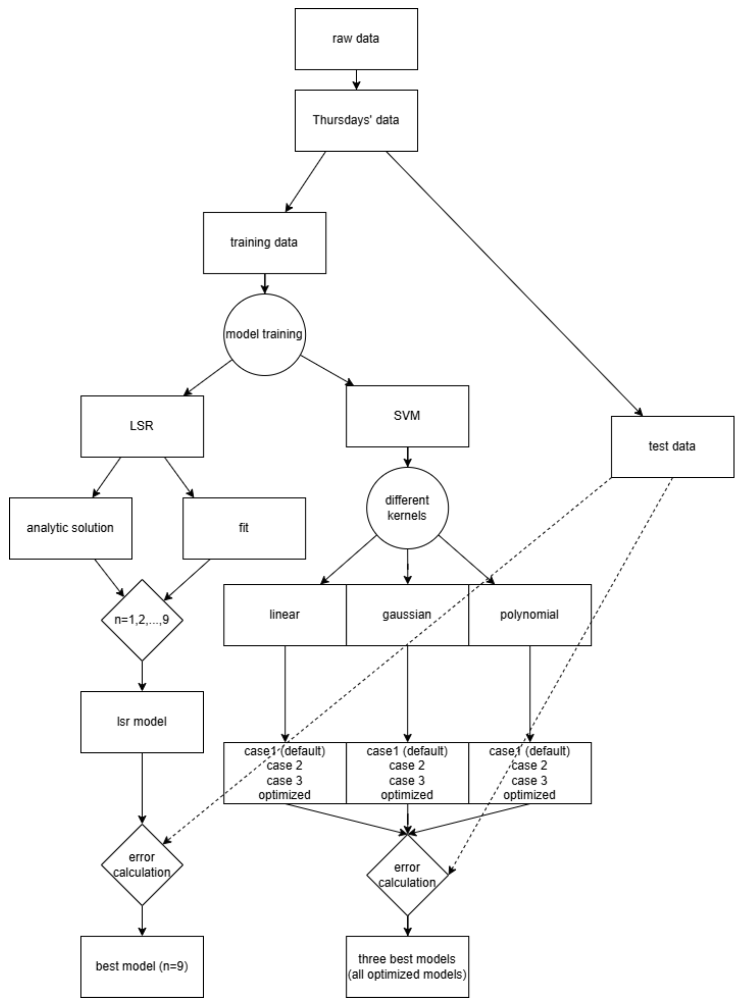

Fig. 1. Enter Caption

For LSR, we employ the analytic solution and the `fit` function in MATLAB. For SVR, we utilize three kernels: linear, Gaussian, and polynomial. Each kernel involves four cases:

1) Default parameters.
2) Two sets of manual parameters.
3) Optimized parameters for the training data.

As for the manual parameters, for the linear kernel:

`BoxConstraint = 10` & `BoxConstraint = 100` for the gaussian kernal:

```
BoxConstraint = 10, KernelScale = 1
 &   BoxConstraint = 100, KernelScale = 10
```

for the polynomial kernal:

```
PolynomialOrder = 5, BoxConstraint = 1000
 &   PolynomialOrder = 9, BoxConstraint = 1000
```

The best model for each kernel is selected based on the Mean Squared Error (MSE) and R^2 of the test data.

The **Mean Squared Error (MSE)** is a measure of the average squared difference between the actual values and the predicted values. It is calculated using the formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where:

- n is the number of data points,
- y_i is the actual observed value for the i-th data point,
- $\hat{y}_i$ is the predicted value for the i-th data point.
- A lower MSE indicates that the model's predictions are closer to the actual values, suggesting better performance.

The **Coefficient of Determination** (**R^2**) evaluates how well the model explains the variability in the observed data. It is given by:

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

where:

- N represents the total number of observations in the dataset,
- $\hat{y}_i$ represents the predicted (or fitted) values from the regression model,
- y_i represents the actual observed values of the dependent variable.

The formula shown represents the coefficient of determination (r²), and here's an explanation of its variables:

- The value of R^2 ranges from 0 to 1:
  - $R^2 = 1$: The model explains all the variance in the data.
  - $R^2 = 0$: The model explains none of the variance, equivalent to a random prediction.
- A higher R^2 indicates that the model better fits the data and captures more variability in the dependent variable.
- However, R^2 alone may not suffice for model evaluation as it does not account for overfitting or model complexity.

B. *Least Square Regression (LSR)*

In polynomial regression, we aim to find a model of the form:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_n x^n$$

where:
- y is the dependent variable (e.g., the total carriageway flow),
- x is the independent variable (e.g., the time in minutes),
- $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients (parameters) to be estimated.

The **Least Squares Regression (LSR)** method is used to estimate these coefficients by minimizing the sum of squared residuals between the observed data and the predicted values. The residuals are the differences between the actual values y_i and the predicted values $\hat{y}_i$:

$$\text{Residual} = y_i - \hat{y}_i$$

The goal is to minimize the sum of squared residuals:

$$J(\beta) = \sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

where m is the number of data points. To solve this, we first expand the data into polynomial terms, leading to the feature matrix X for the training data:

$$X_{\text{train}} = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^n \\ 1 & x_2 & x_2^2 & \ldots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \ldots & x_m^n \end{bmatrix}$$

where each row represents a data point with polynomial terms of x.

The optimal polynomial coefficients $\beta = [\beta_0, \beta_1, \ldots, \beta_n]^T$ are found by solving the normal equation:

$$\beta = (X^T X)^{-1} X^T y$$

where:
- X^T is the transpose of the feature matrix X,
- $X^T X$ is the Gram matrix,
- y is the vector of observed values,
- β is the vector of coefficients to be estimated.

To solve for β, we compute the matrix product $X^T X$, and if the matrix is invertible, we find its inverse $(X^T X)^{-1}$. Multiplying this inverse by $X^T y$ gives the vector of coefficients β.

Once we have the coefficients, we can use them to predict the values of y for any given x. The predicted values $\hat{y}_i$ are calculated as:

$$\hat{y}_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_n x_i^n$$

Alternatively, in MATLAB, we can use the `fit` function in `Curve Fitting Toolbox` to directly perform polynomial fitting:

```
poly_fit = fit(X_train, y_train,
'polyN');)；
```

where N is the polynomial degree, and the function returns the coefficients of the best-fitting polynomial based on least squares regression. This method abstracts away the manual computation of the normal equation.

C. *Support Vector Machines (SVM)*

Support Vector Machines (SVM) are supervised learning models used for both classification and regression tasks. The primary goal of SVM is to find an optimal hyperplane that separates data points into distinct classes, maximizing the margin (the distance between the hyperplane and the closest data points). When the data is not linearly separable, kernel functions are employed to map the input data into a higher-dimensional space where separation becomes feasible.

In this study, SVM models were applied using three types of kernel functions—linear, Gaussian (RBF), and polynomial—on two predesignated datasets. MATLAB's `fitcsvm` function in `Statistics and Machine Learning Toolbox` was utilized for training, with both default parameters and an optimized configuration via `OptimizeHyperparameters`.

a) Key Parameters in SVM: The performance of SVM heavily depends on hyperparameters that control the behavior of the model. Two crucial parameters in MATLAB's `fitcsvm` function are:

1. BoxConstraint (C): This parameter determines the trade-off between margin maximization and misclassification tolerance:

- A larger `BoxConstraint` value penalizes misclassifications more heavily, resulting in narrower margins and higher risk of overfitting.
- A smaller `BoxConstraint` value allows for wider margins and better generalization, albeit at the risk of more misclassifications.

2. KernelScale (σ) for RBF kernels: This parameter adjusts the scale of the Gaussian kernel:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right).$$

A small `KernelScale` results in more sensitive decision boundaries, which may lead to overfitting, while a large `KernelScale` smooths the boundary, potentially underfitting the data. MATLAB can automatically determine an optimal `KernelScale` using:

```
SVMModel = fitrsvm(X, Y,
'KernelFunction', 'gaussian',
'KernelScale', 'auto');
```

b) Kernel Functions: In this study, three kernel functions were explored:

- **Linear Kernel:** Suitable for linearly separable data, with:

$$K(x_i, x_j) = x_i^T x_j.$$

- **Gaussian (RBF) Kernel:** Effective for non-linear relationships, defined as:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right).$$

- **Polynomial Kernel:** Models complex non-linear patterns, defined as:

$$K(x_i, x_j) = (x_i^T x_j + c)^d,$$

  where c is a constant and d is the degree of the polynomial.

c) Optimization in SVM: The optimization process in SVM involves solving a quadratic programming (QP) problem to find the hyperplane that maximizes the margin:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i,$$

subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0,$$

where C (BoxConstraint) controls the trade-off between margin width and misclassification penalties, and ξ_i are slack variables that allow for misclassification.

For this study, MATLAB's `fitrsvm` was used to implement SVM for linear, Gaussian, and polynomial kernels with both default parameters and an optimized configuration. The optimization was achieved using:

```
SVMModel = fitrsvm(X, Y,
'OptimizeHyperparameters', 'auto');
```

d) Summary of Results: By experimenting with different kernels and using both default and optimized settings, the SVM models demonstrated robust classification capabilities. The Gaussian kernel proved particularly effective for non-linear datasets, while the linear kernel performed well for linearly separable data. The results emphasize the importance of carefully selecting kernel functions and tuning hyperparameters such as `BoxConstraint` and `KernelScale` to achieve optimal performance.

III. Experimental Results

A. Time Series and Daily Data

The preprocessed data as time series is shown in Fig. 2. The training data is represented by the blue line, while the test data is shown by the red line. The scatter plot of traffic flow with respect to local time is presented in Fig. 3, where the training data is in blue and the test data is in red.

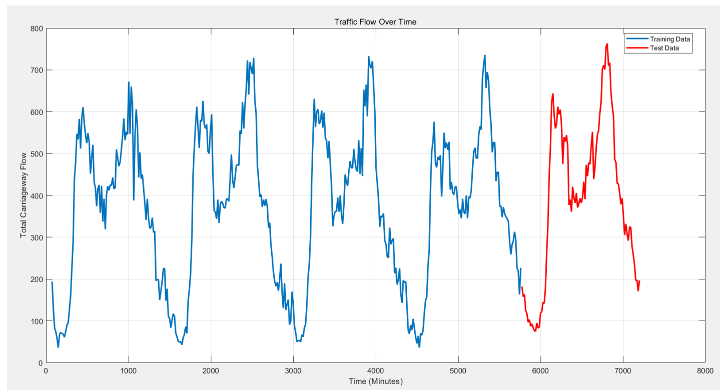

Fig. 2. Time series data with training and test data indicated.

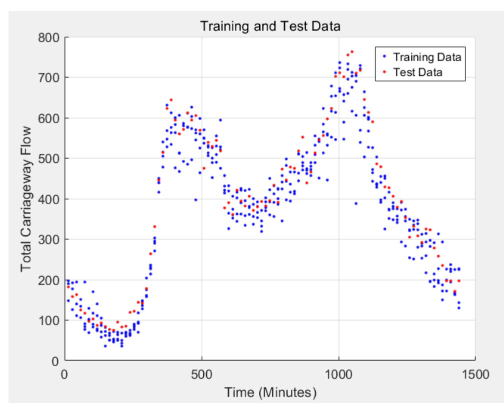

Fig. 3. Scatter plot of traffic flow with local time, showing training and test data.

B. LSR (Least Squares Regression)

Table 1 presents the results of Mean Squared Error (MSE) and R^2 for both the training and test data using the analytic solution, while Table 2 shows the results obtained through the fit function. The results are extremely similar, as both methods rely on the same underlying logic. Figure 4 visualizes these 9 models derived by analytical solution. By comparing the MSE and R^2 values, we observe that as n increases from 1 to 9, the MSE decreases, and the R^2 value approaches 1. In the Fig. 4 is the best fitting model derived using LSR, which is in order of 9. The coefficient for this order 9 model is as follows:

- Coefficient 0 (constant term):
  0.106128760651418796445710768239

- Coefficient 1 (linear term): $9.8322377661486619615516246994 96$
- Coefficient 2 (quadratic term): $-0.16309629802825498234852830137 2$
- Coefficient 3 (cubic term): $0.00101410729791265221549911856 9$
- Coefficient 4 (4th-order term): $-0.00000303554357250287972487079 3$
- Coefficient 5 (5th-order term): $0.00000000500231048356686311772 9$
- Coefficient 6 (6th-order term): $-0.00000000000477912886555986811 5$
- Coefficient 7 (7th-order term): $0.00000000000000262968577009176 3$
- Coefficient 8 (8th-order term): $-0.00000000000000000076978038510 3$
- Coefficient 9 (9th-order term): $0.00000000000000000000009223396 7$

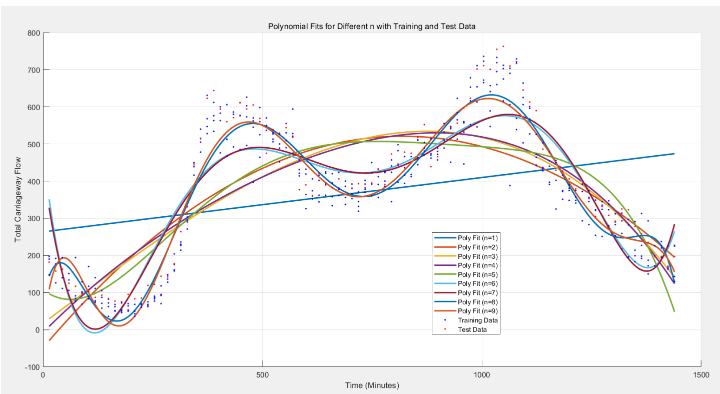

Fig. 4. Polynomial models derived using LSR.

C. SVR (Support Vector Regression)

Tables 3, 4, 5 present the performance of linear, Gaussian, and polynomial kernels in SVM. Each table includes results for four different configurations: the default, two manually set parameters, and one optimized configuration. The settings such as box constraint, kernel scale, and polynomial order are also provided.

D. Model Picking

We choose the best fitting of three SVM models by selecting the model with the smallest MSE and the R^2 closest to 1 (for test data, not the training data) for each kernel. The models chosen are all optimized versions. Figure 5 displays the optimized models of all three kernels along with the best LSR model (n=9), with the scatter plot of the test data shown as blue dots and the training data as red dots. Table 6 shows the MSE and the R^2 of all four optimized model of their own kind.

TABLE VI
PERFORMANCE OF DIFFERENT MODELS

Model	MSE Train	MSE Test	R^2 Train	R^2 Test
LSR	2953.2	3329.7	0.91174	0.90829
SVM_linear	29863.3068	32076.8678	0.1075	0.1165
SVM_gaussian	1593.5792	1407.0586	0.9524	0.9612
SVM_poly	13516.9887	14407.4258	0.5960	0.6032

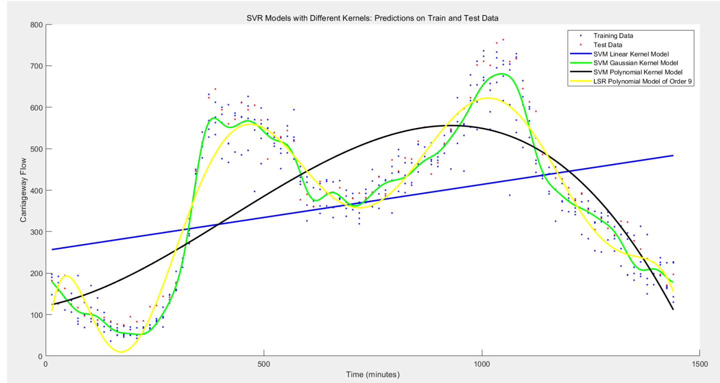

Fig. 5. Best models of LSR and the three optimized MSE kernels

IV. Discussion

This section presents an analysis of the results, focusing on the factors that influence the accuracy of predictions, the limitations of the models used, and potential improvements for future work.

A. Factors Affecting Prediction Accuracy

Several factors contribute to the accuracy of predictions in this study, with the choice of model and kernel being key determinants. The Support Vector Regression (SVR) model, which can handle non-linear data, is sensitive to the selection of the kernel function and the hyperparameters (e.g., C, ϵ, and σ). In this study, a Radial Basis Function (RBF) kernel demonstrated better performance compared to linear and polynomial kernels. The non-linear nature of the RBF kernel allows it to capture more complex relationships in the data, making it more suitable for tasks where patterns are not easily separable by a straight line.

Additionally, the quality of the input data is critical. In this study, features like traffic volume and weather conditions were essential in generating accurate predictions. However, external factors such as data noise, missing values, and the temporal dynamics of the traffic flow can introduce inconsistencies in the model's performance. The accuracy may vary depending on how well these external factors are incorporated into the data preprocessing stage.

B. Model Limitations

Despite the promising performance of the best model—the SVM Gaussian model, which achieved a relatively small MSE of less than 2000 and an R^2 value exceeding 0.95, very close to 1—several limitations remain. The performance of SVR is highly dependent on the choice of kernel. While the RBF kernel delivered strong results for our dataset, datasets with different characteristics might require alternative kernels for optimal performance. Without prior knowledge of suitable parameters for polynomial or linear kernels, some cases exhibited extremely high MSE and significantly negative R^2 values, indicating a fit much worse than the mean. Moreover, this trial-and-error approach is highly time-consuming, especially relative to its efficiency.

Moreover, the limited scope of the training data, particularly when it only captures specific traffic conditions, can impact the model's robustness. For example, if the dataset consists of just four Thursdays from June 2016—amounting to 380 data points in total—it may fail to provide a sufficiently comprehensive representation of the model.

TABLE II
PERFORMANCE OF DIFFERENT ORDERS n IN LSR USING FIT FUNCTION

Model (Fit Function)		n=1	n=2	n=3	n=4	n=5	n=6	n=7	n=8	n=9
Training Error	MSE	29834	13195	12732	12687	11969	6315	6269.7	3080.2	2953.2
	R^2	0.10841	0.60569	0.61952	0.62084	0.6423	0.81128	0.81263	0.90795	0.91174
Test Error	MSE	32158	15085	14313	14152	13453	6852.6	6816.2	3346.1	3329.7
	R^2	0.11421	0.5845	0.60576	0.61019	0.62945	0.81125	0.81225	0.90783	0.90829

TABLE III
PERFORMANCE OF OPTIMIZED LINEAR KERNEL IN SVM

Box Constraint	**MSE Train**	**MSE Test**	R^2 **Train**	R^2 **Test**
1(default)	30688.2250	32443.4562	0.0829	0.1064
10	48918.5234	52827.4914	-0.4619	-0.4551
100	2.9155e+05	3.0685e+05	-7.7129	-7.4521
optimized	29863.3068	32076.8678	0.1075	0.1165

TABLE IV
PERFORMANCE OF GAUSSIAN KERNEL IN SVM

Box Constraint	**Kernel Scale**	**MSE Train**	**MSE Test**	R^2 **Train**	R^2 **Test**
1(default)	default	1455.4431	2043.6282	0.9565	0.9437
10	1000	23934.1123	25859.8781	0.2847	0.2877
100	1000	1676.4521	2471.5314	0.9499	0.9319
optimized	optimized	1593.5792	1407.0586	0.9524	0.9612

TABLE V
PERFORMANCE OF POLYNOMIAL KERNEL IN SVM

Polynomial Order	**Box Constraint**	**MSE Train**	**MSE Test**	R^2 **Train**	R^2 **Test**
3(default)	1(default)	1.3653e+12	1.3511e+12	-4.0801e+07	-3.7215e+07
5	1000	2.0216e+15	2.0216e+15	-6.0416e+10	-5.5685e+10
9	1000	33462	36784	0	-0.013196
optimized	optimized	13516.9887	14407.4258	0.5960	0.6032

C. Potential Improvements

To improve the model's predictive performance, several strategies can be explored. First, expanding the dataset to include a broader range of traffic scenarios and considering other factors into the model could improve the model's ability to generalize. Augmenting the dataset with more diverse data can help reduce overfitting and allow the model to better handle unseen data.

Another potential improvement is the exploration of alternative machine learning models. While SVM performed well, other models such as Random Forest or Gradient Boosting Machines (GBM) might be able to capture the complexity of the data more effectively, especially in the presence of non-linearity. Ensemble methods that combine multiple models could also be considered to improve accuracy and reduce the risk of overfitting. Additionally, neural networks offer the flexibility to model intricate patterns in the data, especially in volatile markets. Both models can enhance prediction accuracy when applied to different aspects of financial forecasting.

Although this study focuses on time series analysis, we opted to use the local time of each day for training and prediction due to the efficiency of the LSR and SVM models. However, in real-world applications, studying the full time series would be more appropriate. In such cases, SARIMA could play a crucial role. SARIMA is particularly effective for capturing seasonality and trends in time series data, making it well-suited for this study's data by leveraging similar daily patterns to improve predictions. On the other hand, neural networks excel at modeling complex, non-linear relationships in large datasets, offering another promising approach for such analyses.

Furthermore, hyperparameter optimization through techniques like grid search or random search could lead to better model performance. Adjusting the C and ϵ parameters in SVM, for instance, can significantly influence the model's ability to fit the data.

Cross-validation should also be employed to ensure that the model is not overfitting and that the results are robust.

D. Conclusion

In conclusion, LSR has problems of not fitting the best, additionally, while SVM is a powerful model for predicting traffic patterns, its performance can be influenced by several factors including the choice of kernel, the quality and scope of the training data, and the potential risk of overfitting. Expanding the dataset, exploring alternative models, and fine-tuning hyperparameters are all strategies that could enhance the model's predictive capabilities. Future research should focus on overcoming the current limitations to improve the generalization of the model to unseen traffic scenarios.

REFERENCES

[1] Highways England. "Highways England WebTRIS." Traffic flow information, June 2016. TMU Site 9956/1 on link A1081 westbound between A505/B653 and M1 J10; GPS Ref: 509008;218723; Westbound. Accessed [date]. http://webtris.highwaysengland.co.uk/..

[2] P. Sun, "MATH 304-Numerical Analysis and Optimization — Intro duction of SVR and Project Part II," Lecture Notes, Duke Kunshan University.