Using regression models to predict stopping distance of a car based on car speed

INTRODUCTION

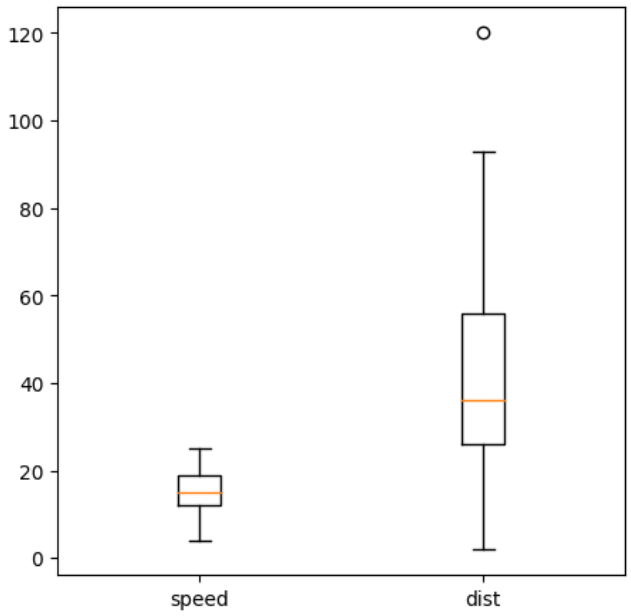
In this paper we will try to solve the problem of predicting a car stopping distance using regression models. The data consists of 50 instances and each instance contains 2 values: the speed that was driven and stopping distance that was measured. After analysing data values, using descriptive analysis one outlier value was found and therefore excluded from the data set.

Nested cross-validation was used for model selection and evaluation to prevent problems that occur when the data set is small. During cross-validation each set of data was split to the training and the test set, then scaled and trained using cross-validation and then evaluated.The models that were selected for this analysis are Linear regression, BayesianRidge, Linear SVM regression, kernelized SVM, Random forest for regression and XGBoost for regression. The model that had the best results is BayesianRidge and that model was chosen for this regression problem.

DATA

Our data set consists of 50 instances and each of them has two features and one target value that presents the value of stopping distance. The goal is to predict the target value based on given feature that represents speed value.

One of the features presents the index of the instance, and was excluded from our data set.

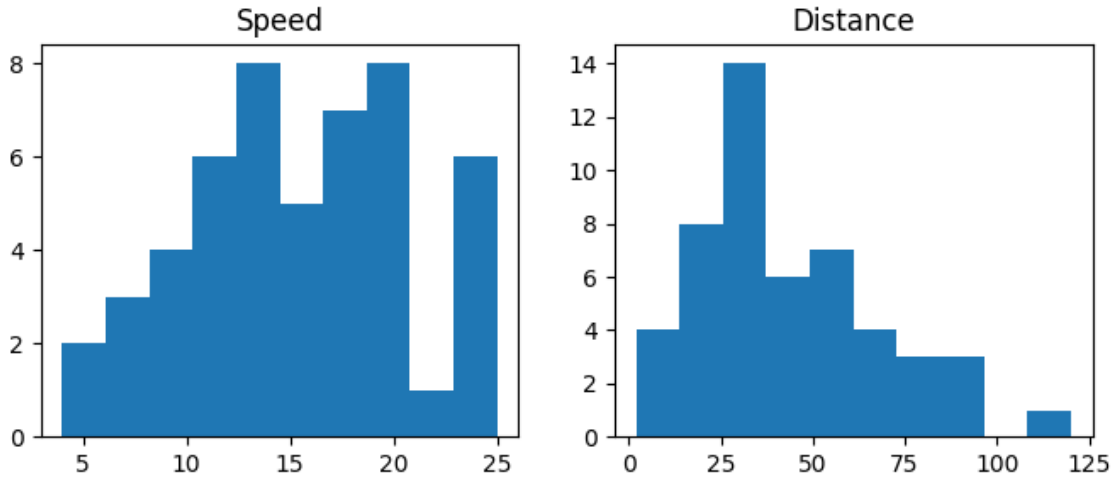


One instance was an outlier and was excluded from the data set.

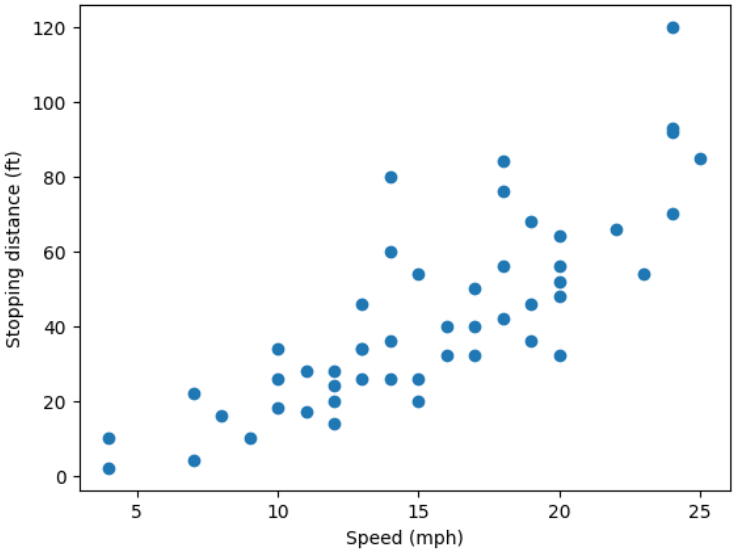
Descriptive analysis of data

	count	mean	std	min	25%	50%	75%	max
speed	50	15.4	5.287644	4	12	15	19	25
dist	50	42.98	25.76938	2	26	36	56	120

Data distribution histogram



The correlation coefficient for feature and target value is 0.8068949006892103.



MODEL SELECTION AND EVALUATION

Nested cross-validation is used on small datasets when testing is problematic and danger of overfitting and it allows us to find the best model and estimate its generalization error correctly.

Models that were trained on data are Linear regression, BayesianRidge, Linear SVM regression, kernelized SVM, Random Forest for regression and XGBoost for regression with different number of estimators and depth.

Hyperparameters used to select models:

XGBoost – max_depth = [2, 3, 4]; n_estimators = [10, 20, 30, 40, 50]

SVM regressor - gamma = [0.001, 0.01, 0.1, 1, 10], C=[10, 100, 1000], kernel=['rbf', 'linear']

Random Forest - max_depth = [2, 3, 4]; n_estimators = [10, 20, 30, 40, 50]

Linear Regression, Linear SVM classifier and BayesianRidge algorithm were trained with no hyperparameters.

RESULTS

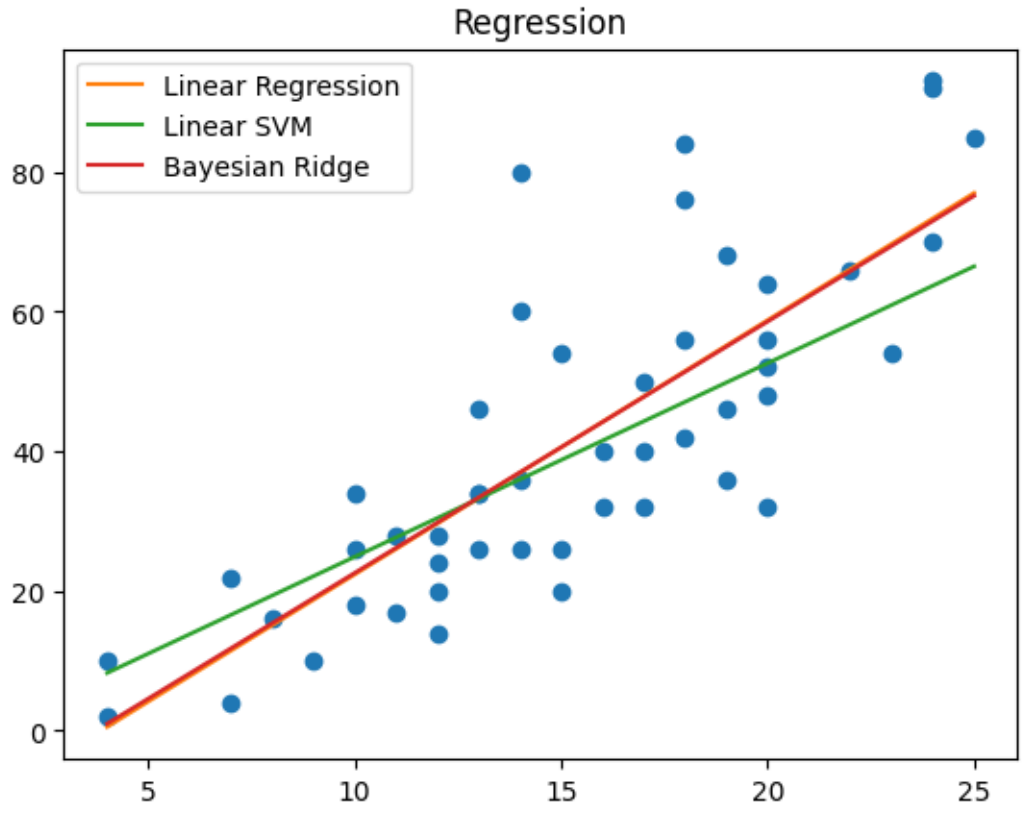
Model	***Best score***	***Params***	***Scores***
SVM	0.79638	C = 1000	0.61988
		gamma = 0.001	0.73434
		kernel = linear	0.64411
			0.79638
			-0.22934
Linear regression	0.80114		0.80114
			0.35889
			0.31706
			0.67890
			0.63204
Linear SVM	0.78958		0.62409
			0.68566
			0.61556
			0.78958
			-0.23359
Bayesian Ridge	0.84892		0.84892
			0.23341
			0.27744
			0.78439

			0.53187
Random Forest	0.73015	max_dept = 2 n_estimators = 20	0.55578 0.73015 0.60325 0.44131 0.65379
XDBoost	0.64416	max_dept = 2 n_estimators = 20	-0.00174 0.46763 -0.22629 0.13135 0.64416

The model that gives best results from all of the trained models is **Bayesian Ridge** so this model will be trained on data and used as estimator.

```
Coefficients = 3.59825993

Intercept = -13.373508373532658
```



Data scatter plot with regression line for different algorithms

CONCLUSION

Even though feature and target data have fairly strong linear correlation, there is some variance in the target data that cannot be explained using feature values without risk of overfitting.