Bike Sharing Volume Predictions

Based on historical weather and rental data from Seoul, South Korea

Wess Kilker
Graduate Programs in Software
University of St Thomas
Saint Paul, MN
kilk4626@stthomas.edu

Tianyu Lei
Graduate Programs in Software
University of St Thomas
Saint Paul, MN
lei39694@stthomas.edu

Jason Xiao
Graduate Programs in Software
University of St Thomas
Saint Paul, MN
xiao4950@stthomas.edu

Jessica Zastoupil
Graduate Programs in Software
University of St Thomas
Saint Paul, MN
zast4939@stthomas.edu

ABSTRACT

Bicycles are available for rent in many large cities. It is important to know how the weather can affect bike rentals so that the city has enough bikes available to meet the rental demand.

This paper looks at weather (Dew Point, Humidity, Rainfall, Snowfall, Solar Radiation, Temperature, Visibility, Windspeed), seasonal information (spring, summer, autumn and winter) and holidays against the number of bikes rented per hour in Seoul, Korea. Four models (Linear Regression, Polynomial Regression, Tree Forest Regression and kNN Regression) were used to assess the Seoul bike dataset, first with all features and then using three separate feature reduction techniques (Backward Elimination, PCA, Kernel PCA), for a total of sixteen different model sets to compare.

Model prediction performance is compared using model accuracy score (coefficient of determination [R^2]), MSE, RMSE, and cross validation mean and standard deviation scores.

The results show that the Tree Forest Regression using all features and 311 estimators produce the highest accuracy model of 88.33% with an MSE of 472.45 and an RMSE of 0.2174. Our cross-validation model accuracy mean is 88.81% with a standard deviation of 0.0137.

CCS CONCEPTS

• Computing Methodologies~Machine learning~Machine learning approaches

KEYWORDS

Bike Sharing, Prediction, Machine Learning

1 Data

The dataset being used for this project was obtained from: https://archive.ics.uci.edu/ml/datasets/Seoul+Bike+Sharing+Demand. The original source of the bike sharing data was: http://data.seoul.go.kr/ which is a public data site maintained by Seoul Metropolitan Government and other organizations in South Korea.

This dataset contains information on bike rentals in Seoul, Korea, between January 12, 2017 and December 11, 2018. The initial number of instances in the dataset is 8760, however we filtered out the non-functioning hours which leaves us with 8465 instances. The dataset is not missing any values.

The dataset contains the count of public bikes rented at each hour from the Seoul Bike Sharing System located in Seoul, South Korea. The attributes of the dataset include:

- Date (day-month-year)
- Rented Bike count (Count of bikes rented at each hour)
- Hour (Hour of the day)
- Temperature (Celsius)
- Humidity (%)
- Windspeed (m/s)
- Visibility (10m)
- Dew point temperature (Celsius)
- Solar radiation (MJ/m2),
- Rainfall (mm)
- Snowfall (cm)
- Seasons (Winter, Spring, Summer, Autumn)
- Holiday (Holiday/No holiday)
- Functional Day (NoFunc(Non-Functional Hours), Func (Functional hours))

The dependent variable in this dataset is the Rented Bike Count. The predictors include everything else except Functional day, which was removed from the list as during initial data preparation.

1.1 Data Preparation

The source data file was not altered prior to import. We imported the data using 'Latin1' encoding as there were special characters in the file that required us to encode the file while importing.

After importing the data several alterations were required. The data was first filtered to remove non-functioning days – days when the bike rental system was not working - as they would not provide any information to predict future rentals. This column was later dropped since having filtered it down to use only functioning days, all rows contained the same value and would make no difference in our predictions.

A "DayOfWeek" feature was added by converting the date to a numeric number (0 – 6), representing Sunday through Saturday to see if the day of the week was a significant factor in bike rentals.

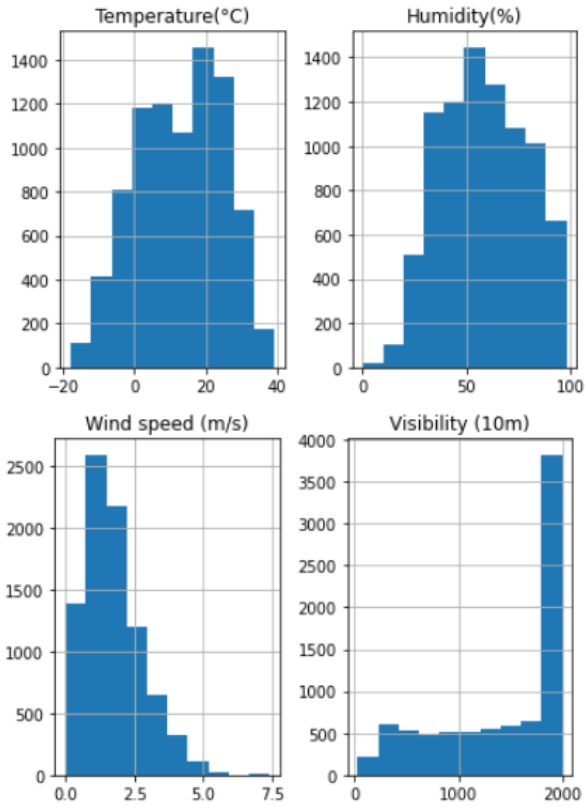
Finally, the unneeded columns Date and Functioning Day were removed, and the dependent variable “Rented Bike Count” was moved to the end since this is what we wanted to use as our dependent y variable. While this is not strictly necessary, it makes the train/test split easier down the road.

Finally, we used one-hot encoding to encode our two categorical variables, which are the seasons and holiday variables. Prior to splitting into training/testing datasets we normalized our continuous data using StandardScaler to be sure that we did not add an unintentional bias due to mismatched ranges of our data points.

For all models, we use a 70/30 split for our training/testing which gave us 5925 instances to train on and 2540 instances to test with.

1.2 Data Exploration

1.2.1 Histograms. Reviewing the histograms of the primary weather features revealed that the Temperature(°C), Humidity (%) and Dew point temperature(°C) features have normal distributions. The Wind speed (m/s), Rainfall(mm) and Snowfall (cm) features are slightly right-skewed, whereas the Solar Radiation (Mj/m2) feature has a more pronounced right-skew. Visibility (10m) was the only left-skewed feature.



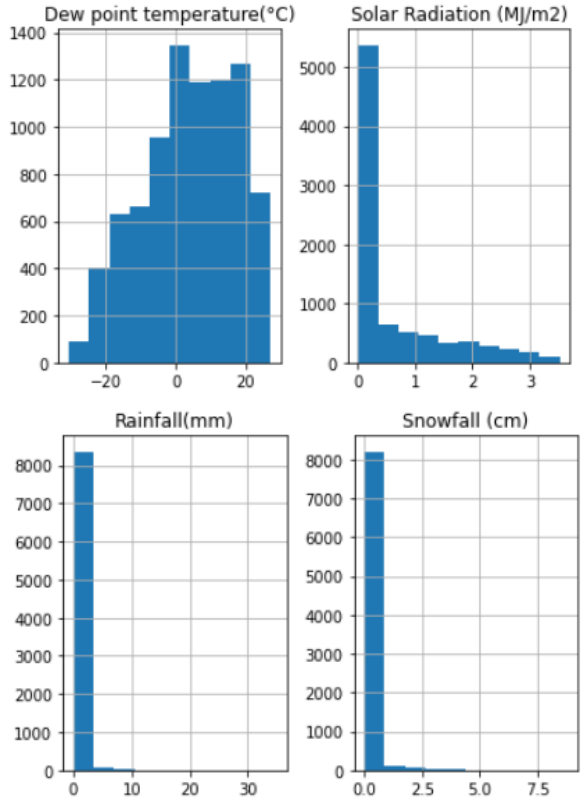


Figure 1: Histograms of Instances for each Weather Feature

1.2.2 Bar Graphs. Reviewing several bar graphs of the dataset grouped by the Seasons and DayOfWeek features, shows that seasonally, as one might expect, bike rentals are highest in the summer. Autumn has slightly more rentals than spring and winter rentals are far lower than the other three seasons.

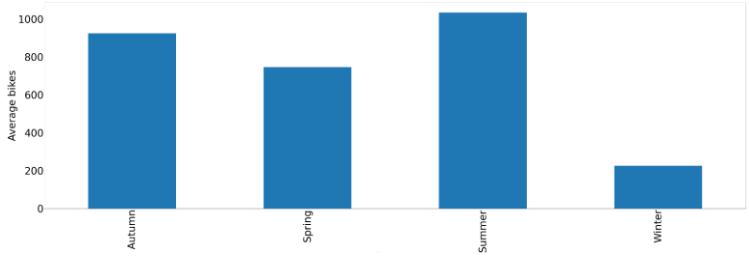


Figure 2: Bar Chart of Average Bike Rentals by Season

2 Feature Selection and Models

We initially took our dataset containing all features and ran the four models against that. The models include Linear Regression, polynomial regression, random tree forest regression and finally, kNN regression. Next, we took the dataset and applied several feature reduction techniques, again running each model against the new dataset with pared down features. The features reduction

techniques we used were Backward Elimination, PCA, and Kernel PCA. This gave us a total of 16 overall model combinations (4 models *4 dataset collections) to choose from in evaluating which model performed best at predicting the bike rentals.

2.1 All Features

First, we applied the four models to the dataset without performing any feature reduction.

2.1.1 Linear Regression with All Features. This gave us an accuracy level of 0.5370, a mean square error (MSE) of 1874.21 and a root mean square error (RMSE) of 0.4329. Our cross-validation model accuracy mean is 0.5398 with a standard deviation of 0.0275.

2.1.2 Polynomial Regression with All Features. 3 runs of polynomial regression with all features and using 2, 3, and 4 degrees, showed that 2 degrees performed the best. Because we must run polynomial using all features prior to splitting we started with the pre-split data from section 1.1. This gave us an accuracy level of 0.6811, a mean squared error (MSE) of 1367.08 and a root mean square error (RMSE) of 0.3697. Our cross-validation model accuracy mean is 0.6963 with a standard deviation of 0.0225.

2.1.3 Tree Forest Regression with All Features. First, we ran a loop between 10 and 2000 with a step of 100 to determine the approximate best number of estimators. This code determined that our best number would be approximately 710, however after approximately 310, the increase in accuracy was insignificant.

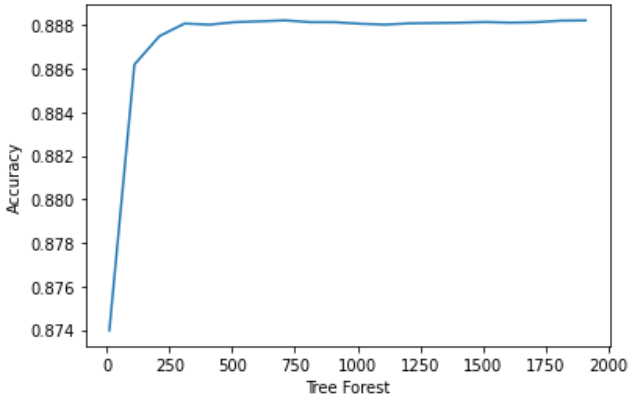


Figure 3: Tree Forest Estimators 10 – 200, Step=100

Therefore, using 310 as our approximate target, we again ran a loop from 305 to 315 with a step of 1 to determine the ideal number of estimators. This code determined that 311 and 312 estimators produced nearly identical results; we chose the lower number of 311 as our target.

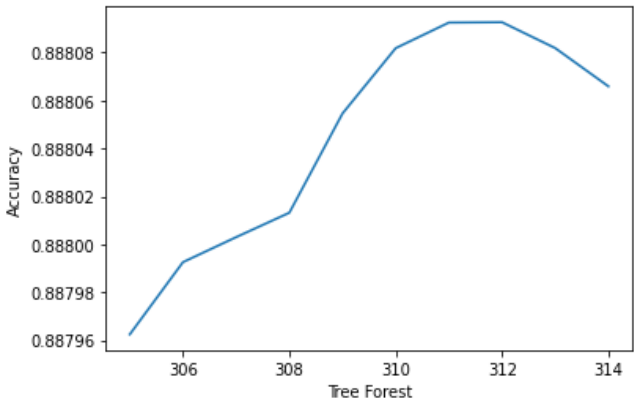


Figure 4: Tree Forest Estimators 305 – 315, Step=1

Setting the n_estimator hyperparameter to 311 gives us an accuracy of 0.8833 with an MSE of 472.45 and an RMSE of 0.2174. Our cross-validation model accuracy mean is 0.8881 with a standard deviation of 0.0137.

2.1.4 kNN Regression with All Features. We ran a loop between 1 and 20 to determine the best number of neighbors for our data, using the Minkowski metric.

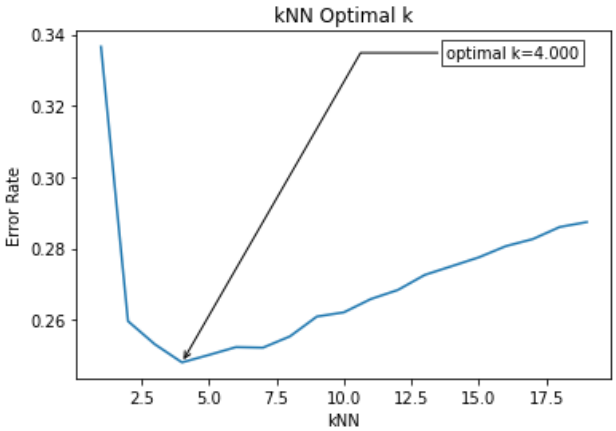


Figure 5: kNN Error Rate, kNN (1-20)

The ideal number of neighbors ended being 4. Running the data through this model gives us an accuracy of 0.7520 with an MSE of 1004.012 and an RMSE of 0.3169. Our cross-validation model accuracy mean is 0.7622 with a standard deviation of 0.0203.

The overall best model on this dataset when using no feature elimination techniques, was the Random Tree Forest Regression with 311 estimators. This gave us an accuracy of .8833.

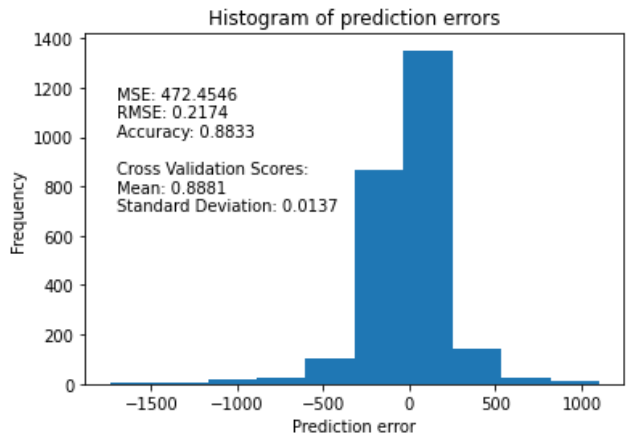


Figure 6: Histogram of Predictions for the Best Model (Tree Forest) using All Features

	Linear Regression	Polynomial Regression	Tree Forest	kNN Regression
Accuracy	0.5370	0.6811	0.8833	0.7520
Cross Val - Mean	0.5398	0.6963	0.8881	0.7622
Cross Val - Std Dev	0.0275	0.0225	0.0137	0.0203
MSE	1874.214	1367.0846	472.4546	1004.0121
RMSE	0.4329	0.3697	0.2174	0.3169

Table 1: Metrics of the Four Models using All Features

2.2 Backward Elimination

Backward elimination is a way of reducing the number of features by starting with all features by fitting a Linear Regression model and removing the predictor with the highest p-value above a certain threshold. This is repeated, removing predictors one at a time until the only predictors left fall below the chosen significance level.

Because the data would be removing features, we chose to start with the preprocessed and normalized dataset we had prior to the split discussed in section 1.1. We used a significance level of .05 to determine features to eliminate with Backwards Elimination. Since this isn't particularly good, we decided to run backward elimination with different significance levels to see if we could get a better accuracy. Interestingly, the overall accuracy didn't change at all when using a significance level of 0.08 or lower and was only very slightly worse when using a significance level of 0.09 or 0.10. All versions removed the seasonal features as being insignificant. Once we got to .08 and below the model also removed the Holiday feature for being insignificant. The significance of the remaining features all stayed the same once we reached 0.08.

After running the full dataset through backwards elimination, we then split the data into training and testing sets at a 70/30 split before generating the four models.

2.2.1 Linear Regression After Backward Elimination. This gave us an accuracy level of 0.5366, a mean squared error (MSE) of 1875.74 and a root mean square error (RMSE) of 0.4331. Our cross-validation model accuracy mean is 0.53798 with a standard deviation of .0264

2.2.2 Polynomial Regression after Backward Elimination. In order to find the best number of degrees to use for polynomial regression, we ran a loop of degrees which determined that our best degree to use would be a degree of 3 with an RMSE of 298.86. Because we had to run the polynomial features code prior to splitting we once again started with the pre-split data from section 1.1. This gave us an accuracy level of 0.7793, a mean squared error (MSE) of 893.20 and a root mean square error (RMSE) of 0.2989. Our cross-validation model accuracy mean is 0.7415 with a standard deviation of 0.051.

2.2.3 Tree Forest Regression after Backward Elimination. First, we ran the Tree Forest Regression code using default hyperparameters. This gave us an accuracy of 0.8554 with an MSE of 585.12 and an RMSE of 0.2419.

Next, we ran a loop between 10 and 2000 with a step of 10 to determine the best number of estimators. This code determined that our best number would be 360. Using this as our n_estimator hyperparameter gives us an accuracy of 0.8559 with an MSE of 583.38 and an RMSE of 0.2415, which is only slightly better than the default numbers we ran earlier for Tree Forest Regression. Our cross-validation model accuracy mean is 0.8580 with a standard deviation of 0.0133.

2.2.4 KNN Regression after Backward Elimination. First, we ran a loop between 1 and 20 to determine the best number of neighbors for our data, using the Minkowski metric. The number of neighbors we ended up using is 6. Running the data through this model gives us an accuracy of 0.8097 with an MSE of 770.105 and an RMSE of 0.2775. Our cross-validation model accuracy mean is 0.81321 with a standard deviation of 0.0156.

The overall best model on this dataset when using Backward elimination for feature selection ended up being the Random Tree Forest Regression with 360 estimators which gave us an accuracy score (R^2) of .8558.

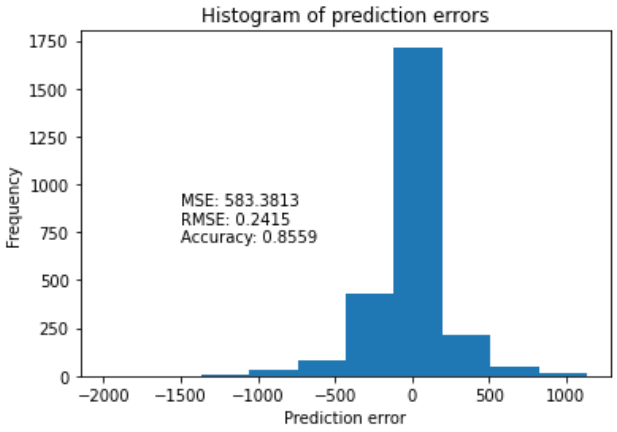


Figure 7: Histogram of Predictions for the Best Model (Tree Forest) after Backwards Elimination

	kNN Regression	Linear Regression	Polynomial Regression	Tree Forest
Accuracy	0.8097	0.5366	0.7793	0.8559
Cross Val - Mean	0.8132	0.5380	0.7415	0.8580
Cross Val - Std Dev	0.0156	0.0264	0.0514	0.0133
MSE	770.1050	1875.7433	893.2027	583.3813
RMSE	0.2775	0.4331	0.2989	0.2415

Table 2: Metrics of the Four Models after Backwards Elimination

2.3 Regression After PCA

Principal Component Analysis (PCA) used for unsupervised dimensionality reduction by projecting the data in the direction of largest variance. It is a method that rotates the dataset in a way such that the rotated features are statistically uncorrelated. This rotation is often followed by selecting only a subset of the new features, according to how important they are for explaining the data.

After we normalized the data and split it to training and test set at a ratio of 70/30. We apply the PCA and fit the logistic regression to the training set to determine the ideal number of components to keep. Then we build the four models using the PCA transformed data and make the prediction on the PCA transformed test set. Next, we calculated the model accuracy for the training set and test set. Then, we run cross-validation model at the end of different regression to have more accurate results.

2.3.1 Linear Regression after PCA. This gave us an accuracy level of 0.5370, a mean squared error (MSE) of 1875.214 and a root mean square error (RMSE) of 0.4329. And cross-validation model accuracy mean is 0.5398 with a standard deviation of 0.0275.

2.3.2 Polynomial Regression after PCA. This gave us an accuracy level of 0.6357, a mean squared error (MSE) of 1474.5398 and a root mean square error (RMSE) of 0.384. And cross-validation model accuracy mean is 0.6424 with a standard deviation of 0.0565.

2.3.3 Tree Forest Regression after PCA. This gave us an accuracy level of 0.7750, a mean squared error (MSE) of 910.7867 and a root mean square error (RMSE) of 0.3018. And cross-validation model accuracy mean is 0.7786 with a standard deviation of 0.0216.

2.3.4 KNN Regression after PCA. This gave us an accuracy level of 0.7520, a mean squared error (MSE) of 910.7867 and a root mean square error (RMSE) of 0.3018. And cross-validation model accuracy mean is 0.7622 with a standard deviation of 0.0203.

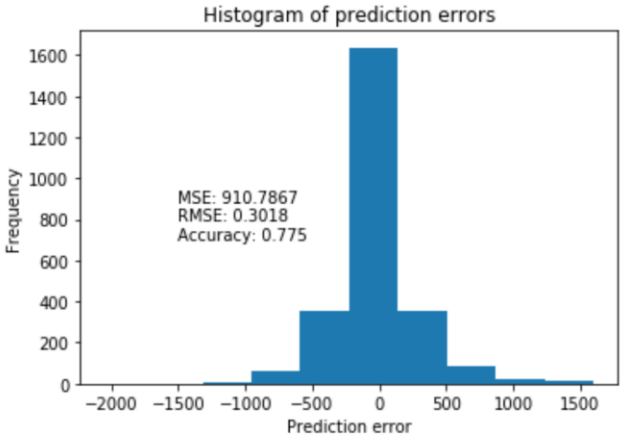


Figure 8: Histogram of Predictions for the Best Model (Tree Forest) using PCA

	kNN Regression	Linear Regression	Polynomial Regression	Tree Forest
Accuracy	0.7520	0.5370	0.6357	0.7750
Cross Val - Mean	0.7622	0.5398	0.6424	0.7786
Cross Val - Std Dev	0.0203	0.0275	0.0565	0.0216
MSE	910.7867	1875.214	1474.5398	910.7867
RMSE	0.3018	0.4329	0.384	0.3018

Table 3: Metrics of the Four Models using PCA

2.4 Kernel PCA

Kernel PCA is an extension of PCA that uses techniques of kernel methods. Application of the kernel PCA transforms data that is not linearly separable onto a new, lower dimensional subspace that is suitable for linear classifiers. Applying kernel techniques creates the ability to tackle nonlinear problems by projecting them

onto a new feature space of higher dimensionality where the classes become linearly separable. Nonlinear combinations of the original features are created in order to map the original dimensional dataset onto a larger dimensional feature space. Kernel PCA has previously been demonstrated to be generally useful for novelty detection and image de-noising.

The same process for PCA was repeated for Kernel PCA. We normalized the data and split it to training and test set at a ratio of 70/30. We apply the Kernel PCA and fit the logistic regression to the training set to determine the ideal number of components to keep. Then we build the four models using the Kernel PCA transformed data and make the prediction on the Kernel PCA transformed test set. Next, we calculated the model accuracy for the training set and test set. Then, we run cross-validation model at the end of different regression to have more accurate results.

2.4.1 Linear Regression after Kernel PCA. This gave us an accuracy level of 0.5705, a mean squared error (MSE) of 1738.3417 and a root mean square error (RMSE) of 0.4169, and cross-validation model accuracy mean is 0.5911 with a standard deviation of 0.0285.

2.4.2 Polynomial Regression after Kernel PCA. This gave us an accuracy level of 0.7013, a mean squared error (MSE) of 1209.0369 and a root mean square error (RMSE) of 0.3477, and cross-validation model accuracy mean is 0.6506 with a standard deviation of 0.1241.

2.4.3 Tree Forest Regression after Kernel PCA. This gave us an accuracy level of 0.7354, a mean squared error (MSE) of 1070.978 and a root mean square error (RMSE) of 0.3273, and cross-validation model accuracy mean is 0.7388 with a standard deviation of 0.0164.

2.4.4 KNN Regression after Kernel PCA. This gave us an accuracy level of 0.7206, a mean squared error (MSE) of 1070.978 and a root mean square error (RMSE) of 0.3273, and cross-validation model accuracy mean is 0.7332 with a standard deviation of 0.0245.

The best model for the dataset after Kernel PCA ended up being Random Tree Forest Model. We achieved an accuracy level of 0.7354 which was marginally higher than kNN Regression and Polynomial Regression.

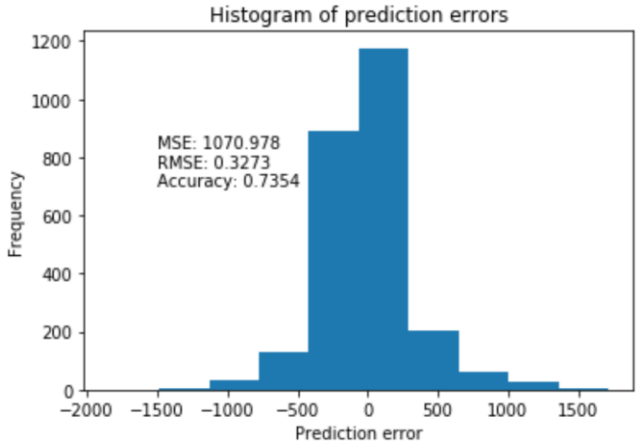


Figure 8: Histogram of Predictions for the Best Model (Tree Forest) using Kernel PCA

	kNN Regression	Linear Regression	Polynomial Regression	Tree Forest
Accuracy	0.7206	0.5705	0.7013	0.7354
Cross Val - Mean	0.7332	0.5911	0.6506	0.7388
Cross Val - Std Dev	0.0245	0.0285	0.1241	0.0164
MSE	1070.978	1738.3417	1209.0369	1070.978
RMSE	0.3273	0.4169	0.3477	0.3273

Table 4: Metrics of the Four Models using Kernel PCA

3 Interpretations and Conclusions

Model	*Elimination Type*	Accuracy
Tree Forest	All Features	0.8833
	Backward Elimination	0.8559
	PCA	0.7750
	kPCA	0.7354
kNN Regression	Backward Elimination	0.8097
	PCA	0.7520
	All Features	0.7520
	kPCA	0.7206
Polynomial Regression	Backward Elimination	0.7793
	kPCA	0.7013
	All Features	0.6811
	PCA	0.6357
Linear Regression	kPCA	0.5705

Model	*Elimination Type*	Accuracy
	PCA	0.5370
	All Features	0.5370
	Backward Elimination	0.5366

Table 5: Accuracy of all 16 Models, Sorted by Overall Model Accuracy and then Feature Set Accuracy

Regardless of feature reduction technique, the order of performance of the four models remained consistent with Tree Forest Regression performing the best, followed by kNN, Polynomial, and Linear Regression respectively. Further interpretation of the models.

When looking further into the results of Linear Regression, only the Kernel PCA transformation improved accuracy of the Linear Regression model, contrasting the PCA transformation which did not change the accuracy and Backward Elimination which lead to a decrease in accuracy as compared to the accuracy of All Features.

Furthermore, when looking into the results of Polynomial Regression, in comparison to our baseline accuracy of All Features (degrees = 2), Backward Elimination (degrees = 3) outperformed the other feature reduction techniques. Kernel PCA (degrees = 2) transformation also improved the accuracy, but not to the degree of Backward Elimination. PCA (degrees = 2) transformation led to a decrease in accuracy value.

Overall, kNN Regression performed the second best in terms of the four models. All Features (kNN = 4) resulted in an accuracy of 75.20% which we used as a baseline for comparison. The accuracy of Backward Elimination (kNN = 6) performed the best. PCA (kNN = 4) did not change accuracy. Kernel PCA (kNN = 4) result in a decrease in accuracy.

The Random Tree Forest Regression model performed the best overall. Interestingly, All Features resulted in the highest accuracy value, while all other feature reductions led to a decrease in accuracy in comparison to it. Backward Elimination resulted in a marginal decrease in accuracy while PCA and kPCA followed respectively.

After the building all the models, we determined that the model with the highest accuracy was the Tree Forest Regression with 311 estimators model using all features, which has an accuracy of 88.33% with an MSE of 472.45 and an RMSE of 0.2174. Our cross-validation model accuracy mean is 88.81% with a standard deviation of 0.0137.

REFERENCES

[1] Sathishkumar V E, Jangwoo Park, and Yongyun Cho. 'Using data mining techniques for bike sharing demand prediction in metropolitan city.' Computer Communications, Vol.153, pp.353-366, March, 2020

[2] Sathishkumar V E and Yongyun Cho. 'A rule-based model for Seoul Bike sharing demand prediction using weather data' European Journal of Remote Sensing, pp. 1-18, Feb, 2020