

Regression Models to Predict the Bio-activities of Different Compounds from the SARS-COVID 19 Database of ChEMBL

Name : Karthik Babu Nambiar

Roll Number : 17120

Here we are trying to predict the bio activities of different compounds from the SARS-COVID 19 database of ChEMBL. We have fitted the data to five different models and evaluated the fitting of the data in the model using the root mean squared error. We download the database from <https://www.ebi.ac.uk/chembl/> in csv format. Then we store this data as pandas DataFrame.

There are 6900 samples in the dataset with 30 features and corresponding 6900 labels (Bio-activities).

Data Cleaning

Then we go on to clean the dataset.

We drop 15 features that have very low correlation with the labels and also features which have Nan or None entry. We also drop all samples that have molecular weights less than 500. We have also dropped a sample which was an outlier. After cleaning the dataset, we have got 5655 samples with 15 features and 5655 corresponding labels.

Helper Functions

Next we go on to define some helper functions that does data splitting, fitting and visualizing much easier.

split(test_size = 0.2, poly_degree = 1)

This function the dataset after data cleaning and splits the data into four sets using the parameter `test_size`: `training_samples`, `training_labels`, `testing_samples`, `testing_labels` using the function `train_test_split()` that Scikit-Learn provides. This function also creates polynomial degrees using the `PolynomialFeatures()` function of Scikit-Learn. The degree of the polynomial features is chosen using the parameter `poly_degree`. This function also scales the dataset using the `StandardScaler()` function of Scikit-Learn.

Parameters

test_size : float or int, default=0.2

If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the test split. If int, represents the absolute number of test samples.

poly_degree : int, default=1

The degree of the polynomial features.

model_score(model, data)

This function outputs the root mean squared error of the predicted values to the true values of the training set and test set. This function also outputs the mean of the 5-fold cross validation score done on the training set using the

`cross_val_score()` of Scikit-Learn.

Parameters

model : Scikit-Learn model with .predict() method

Eg. `sklearn.linear_model.LinearRegression()`

data: python dictionary containing X_train, y_train, X_test, y_test

Output of `split()`

plot_train_prediction(model, data)

This function outputs the scatter plot of predictions made by the

`model.predict()` method in the testing samples (y-axis) vs the true testing labels. This function also plots an orange $y=x$ line to show the fitting line in case of a perfect fit.

Parameters

model : Scikit-Learn model with .predict() method

Eg. `sklearn.linear_model.LinearRegression()`

data: python dictionary containing X_train, y_train, X_test, y_test

Output of `split()`

plot_learning_curve(model, data)

This function plots the root mean squared error of both the training data and the cross validation data vs different training set size using the `learning_curve()` function of Scikit-Learn.

Parameters

model : *Scikit-Learn model with .predict() method*

Eg. `sklearn.linear_model.LinearRegression()`

data: python dictionary containing **X_train, y_train, X_test, y_test**

Output of `split()`

plot(model_name, max_degree)

This function generates `plot_train_prediction()`, `plot_test_prediction()`, `plot_learning_curve()` for a model trained in `max_degree` polynomial features.

Parameters

model_name : *str*

name of the model to be plotted

max_degree: *int*

The maximum degree of polynomial features.

Models

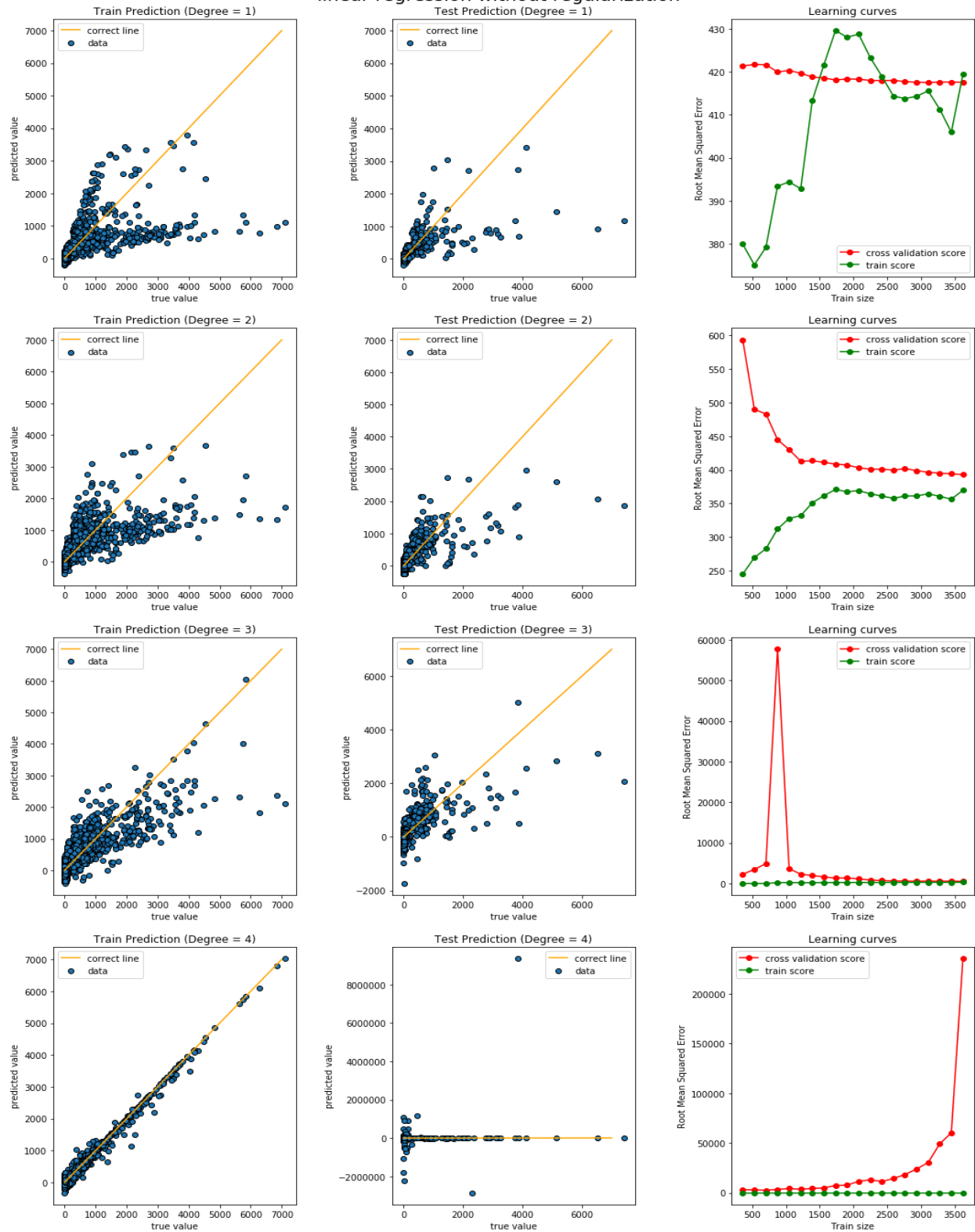
Here we fit the data (with polynomial degrees upto 4 degrees) into five different models and compare their scores.

- Linear regression without regularization
- Linear regression with regularization or Ridge regression
- Decision tree
- Bayesian Regression

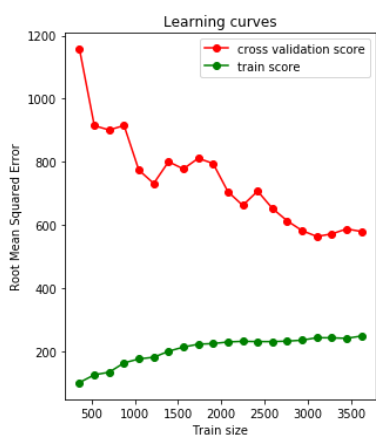
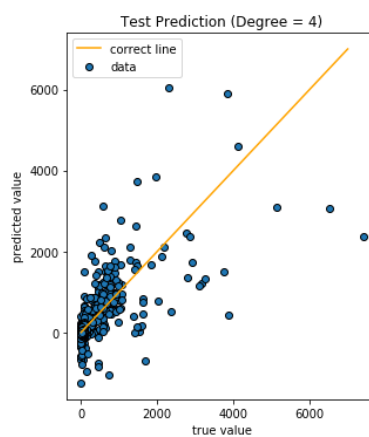
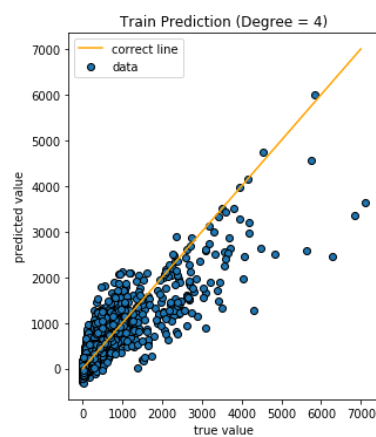
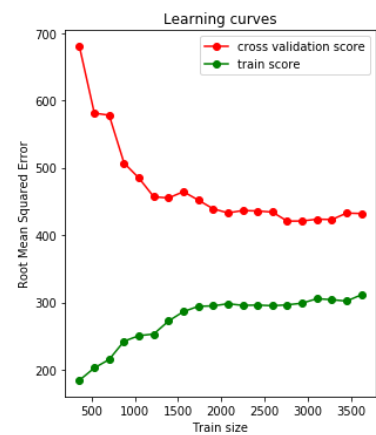
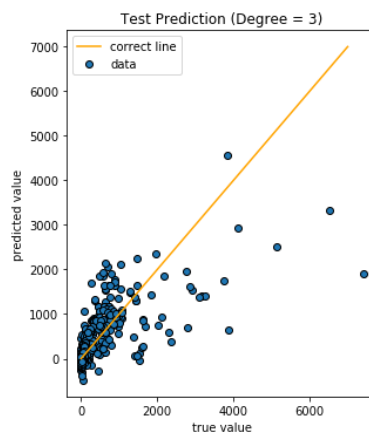
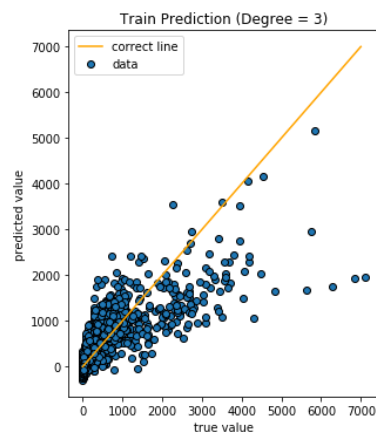
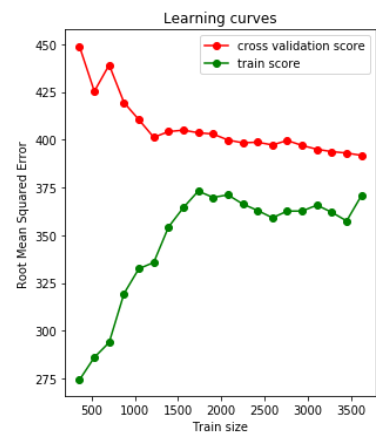
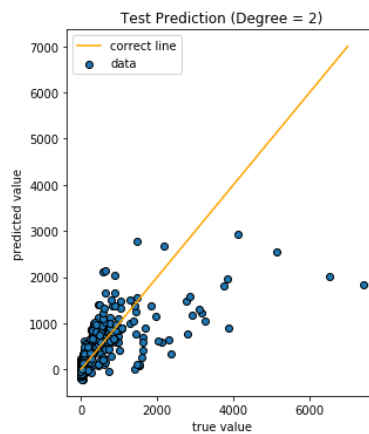
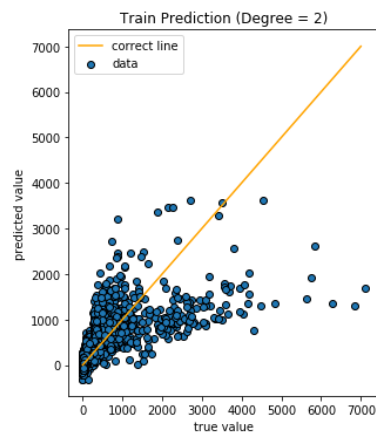
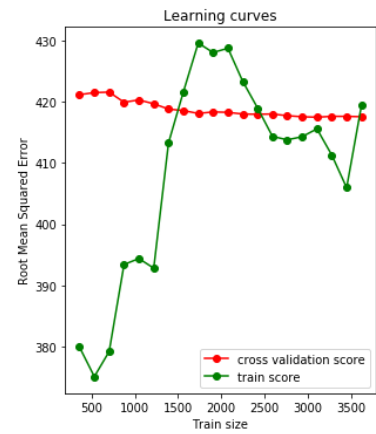
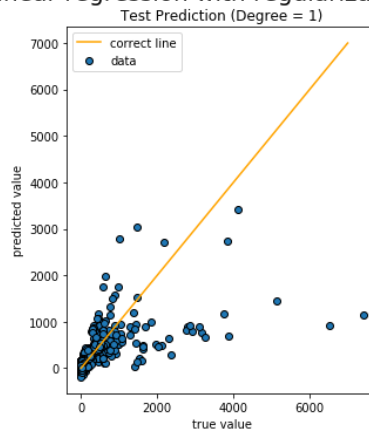
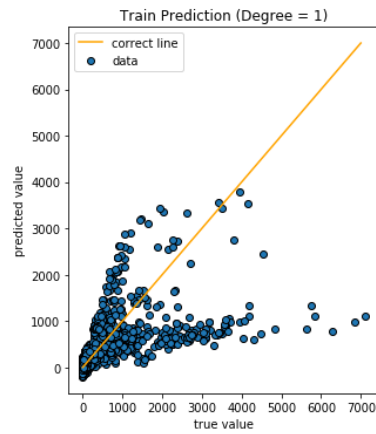
Training fit, testing fit and the learning curve has been plotted for all five models fitted in upto 4 degrees of polynomial features.

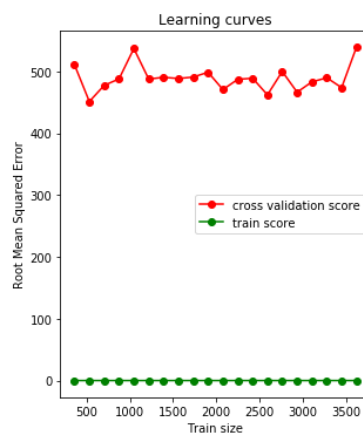
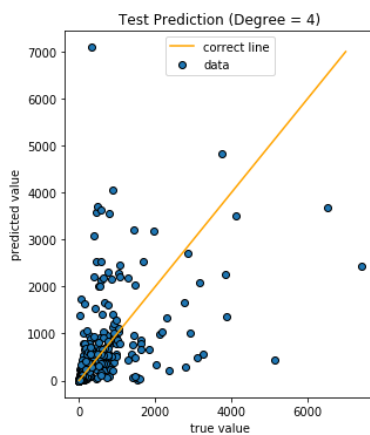
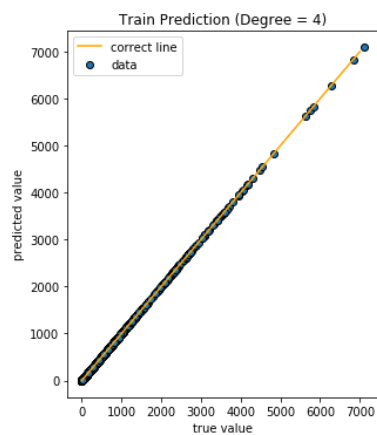
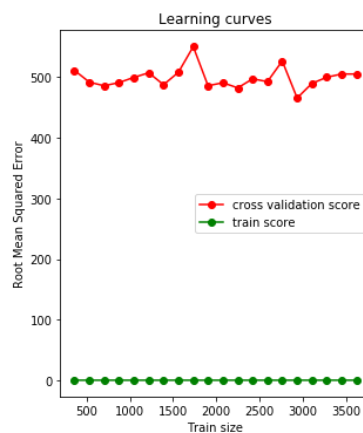
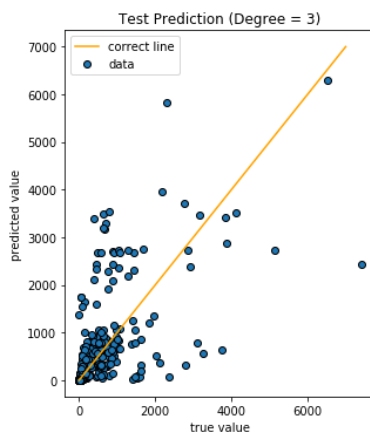
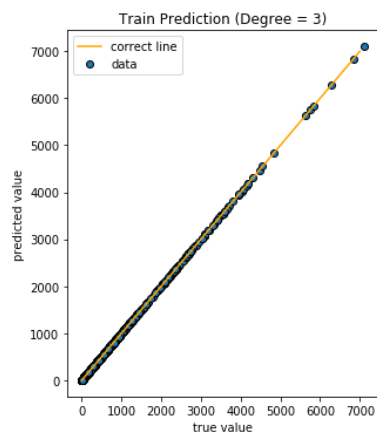
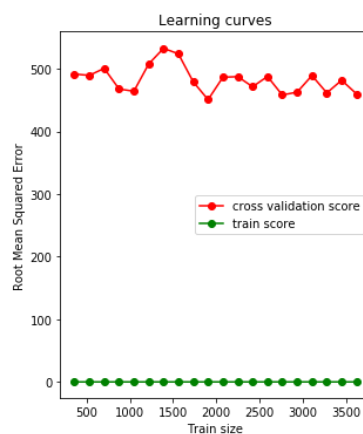
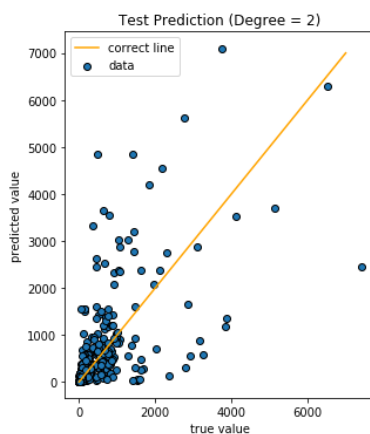
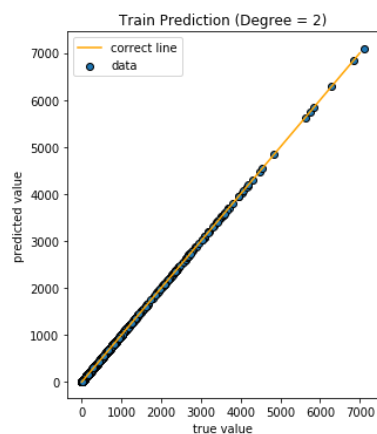
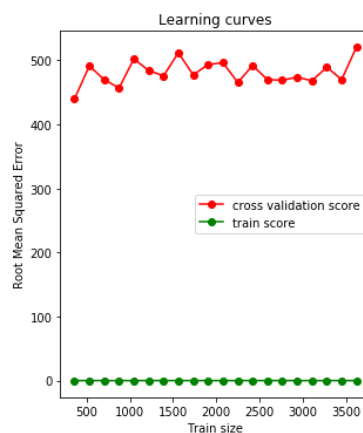
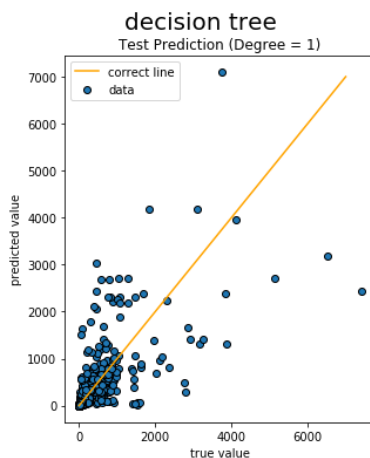
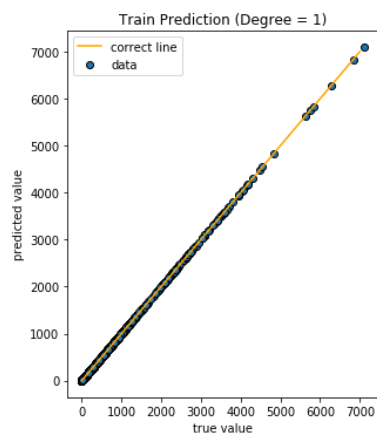
Results

linear regression without regularization

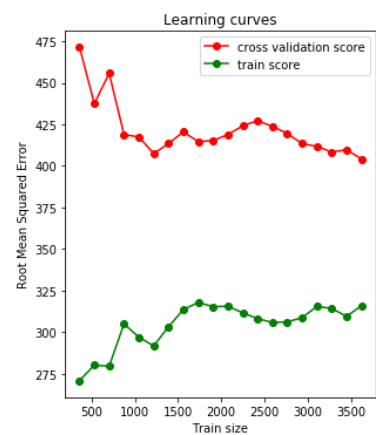
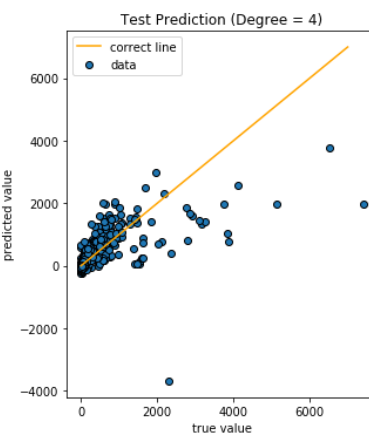
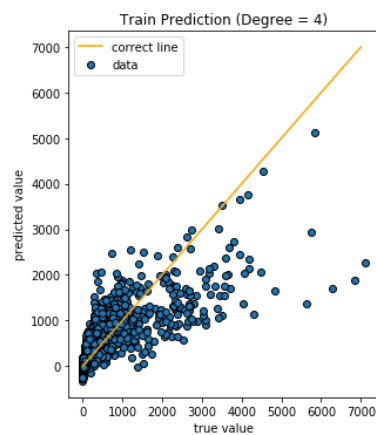
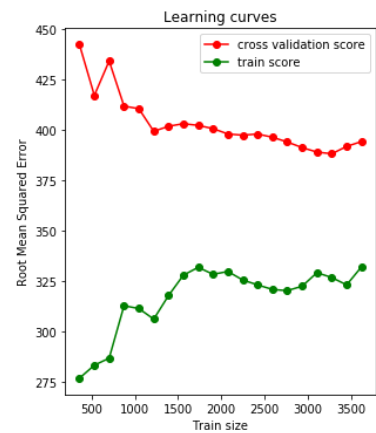
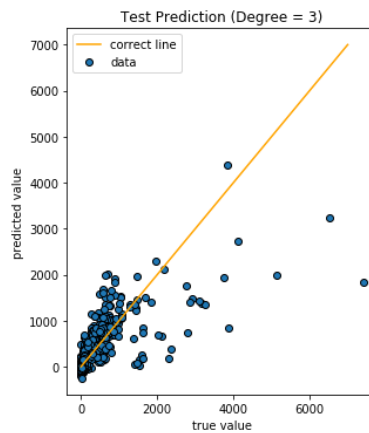
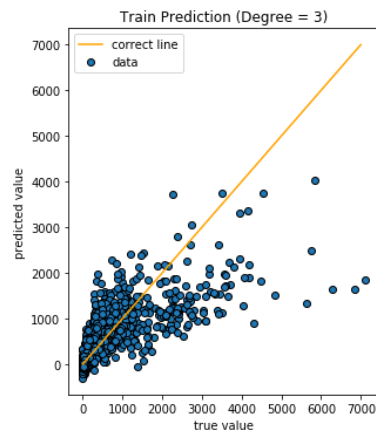
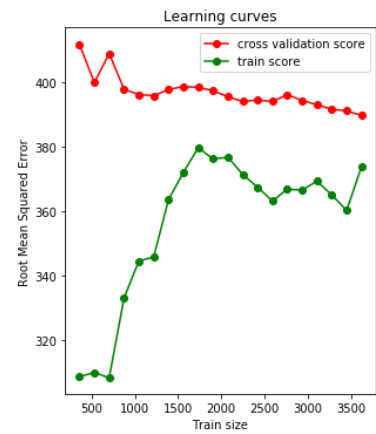
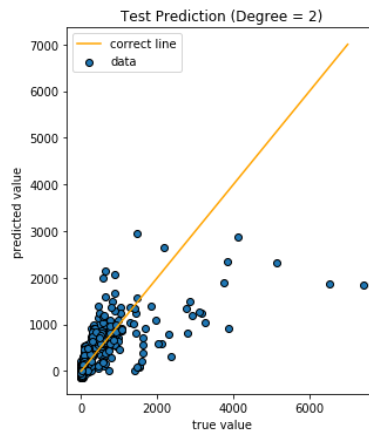
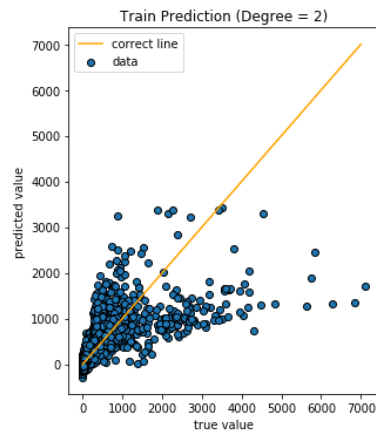
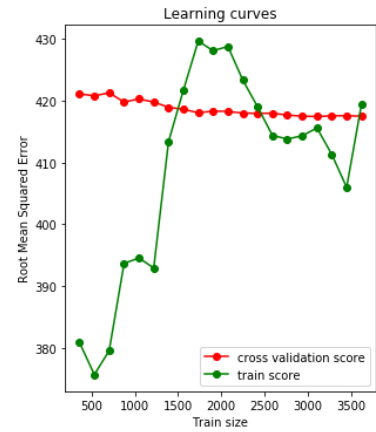
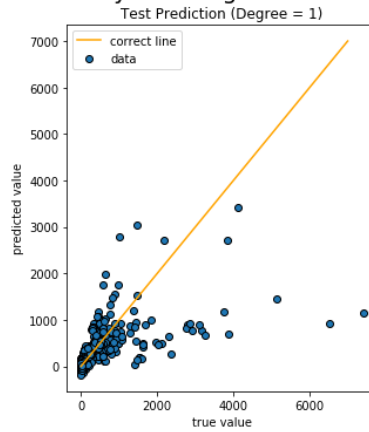
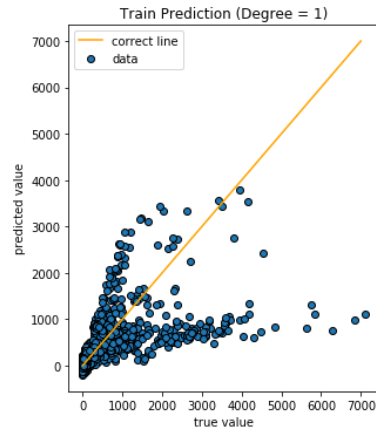


linear regression with regularization





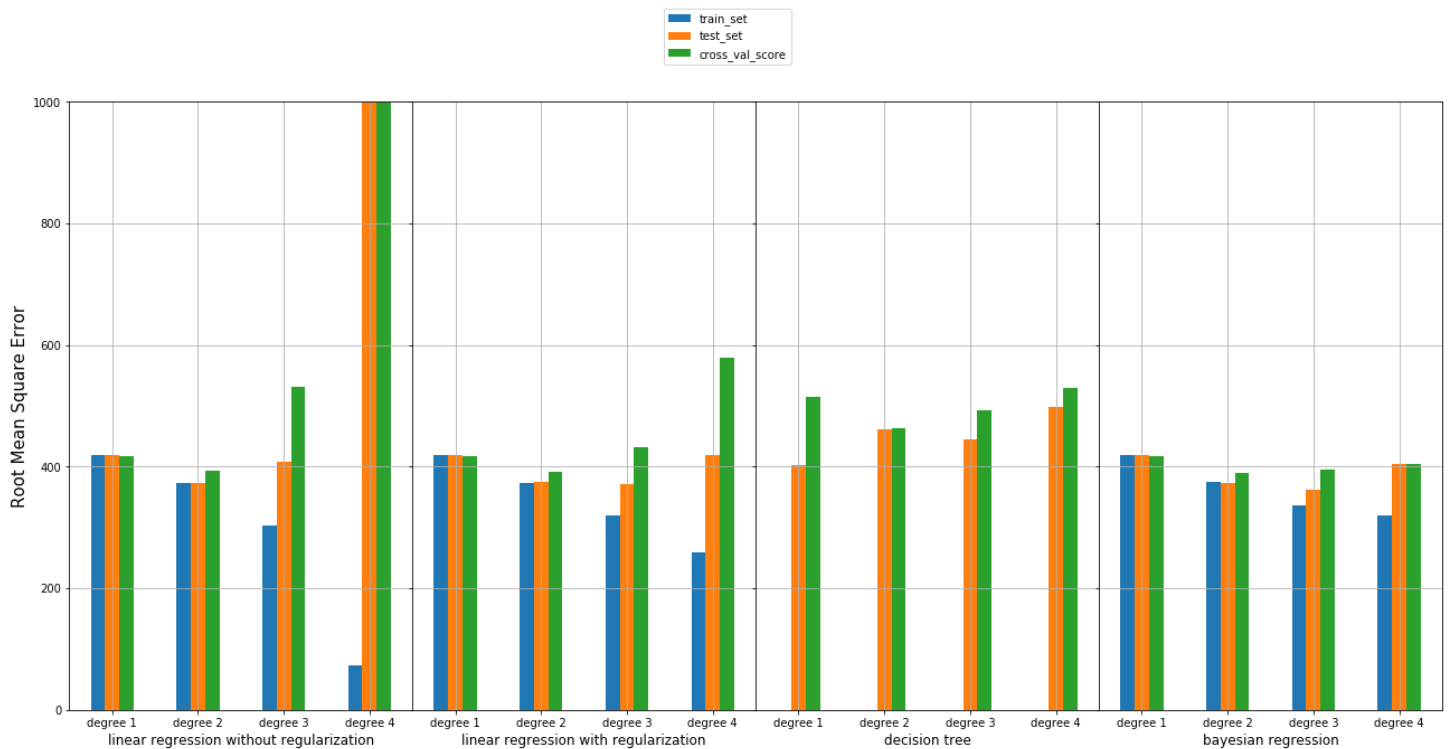
bayesian regression



Scores of different models:

| Model | Degree | train_set | test_set | cross_val_score |
|--|----------|------------|---------------|-----------------|
| Linear Regression Without Regularization | degree 1 | 419.866307 | 418.615324 | 417.576640 |
| | degree 2 | 372.359705 | 373.617408 | 392.908943 |
| | degree 3 | 302.323809 | 408.804532 | 530.672084 |
| | degree 4 | 73.435118 | 315326.589993 | 234891.759073 |
| Linear Regression With Regularization | degree 1 | 419.866345 | 418.620364 | 417.572391 |
| | degree 2 | 373.331604 | 374.237336 | 391.771967 |
| | degree 3 | 319.505548 | 371.551272 | 431.990171 |
| | degree 4 | 259.533636 | 419.349684 | 579.059199 |
| Decision Tree | degree 1 | 0.010513 | 402.316586 | 515.481392 |
| | degree 2 | 0.010513 | 460.950956 | 463.807661 |
| | degree 3 | 0.010513 | 445.487777 | 491.780578 |
| | degree 4 | 0.010513 | 499.054130 | 529.658289 |
| Bayesian Regression | degree 1 | 419.876959 | 418.721758 | 417.520349 |
| | degree 2 | 375.652153 | 372.505208 | 389.671285 |
| | degree 3 | 337.025040 | 362.046311 | 394.414341 |
| | degree 4 | 320.420003 | 403.531846 | 404.180608 |

A bar plot of the root mean squared error of all the models fitted to 4 degrees of polynomial features has been plotted to compare the fitting of the data.



Conclusions

While comparing different model scores we observe that

- **Overfitting** is seen in **linear regression without regularization** for higher degrees of polynomial features as there is low training set error and high cross validation and test set errors.
- **Overfitting** is also observed in **Decision Tree** as there is a significant difference between the training set error and cross validation and test set errors even though they are comparable with other models cross validation and test set errors.
- **Bayesian Regression** and **Ridge Regression** gives a good score in all training set, test set and cross validation set.