**Project Report**

**Project:**

In this project work it was required to apply existing methods to the data for analysis, select the best parameters to increase the accuracy of the model, and apply the selected models to the new test data, which represent images without known centers and predict the values for them. Major tasks were completed, and the results are given below.

**Model Selection and Model Assessment:**

Firstly, we implemented Linear Regression in order to observe the behavior of the data and measured the accuracy on train dataset, which gave us mean squared error (MSE) equal to 0.106 and R squared value(R2) of 0.409. Implemented source code and visualization are written to the attached files, refer to “Source code.ipynb” .

Then we tried KNN and Decision tree methods. For KNN method with 2 neighbors we got results of MSE equals to 0.048 and R2 equals to 0.731 on training dataset. For Decision tree method we achieve results of MSE equals to 0 and R2 equals to 1. Implemented source codes and visualizations are written to the attached files, refer to in other methods folder “KNN and Decision Tree.ipynb”. The major problem that we observe was an overfitting the train dataset by two of these methods and we decided to not use these methods prediction results for the project. Since there are a lot of images of similar in some parts symbols and these methods can just apply the center of train dataset symbols to test.

As for CNN methods we tried to build our own CNN and recreated AlexNet CNN, which was rebuild for our training dataset. The AlexNet is computationally costly method and took more than 15 minutes to complete 1 epoch. This method took more than 5 hours to build CNN for training dataset with 4 folds and 4 epochs and the performance of AlexNet on the training dataset are: MSE equals to 0.296 and R2 equals to -0.662. The issue of AlexNet was not implementing sigmoid activation method in order to get output values between 0 and 1. However, due to costly implementation the sigmoid activation method was applied for CNN. Implemented source codes and visualizations of AlexNet are written to the attached files, refer to in other methods folder “Alexnet.ipynb”.

Our own build CNN model has all activation method “relu” and the last layer with “sigmoid” activation methos. Our model is much faster and for whole training dataset with 4 folds and 10 epochs took several minutes to build prediction of neural network. Corresponding MSE was 0.121 and R2 equals to 0.321. The both metrics can be increased by increasing the number of epochs; however, we can deal with overfitting issue. Implemented source codes and visualizations of our own CNN are written to the attached files, refer to in other methods folder “CNN.ipynb”.

Moreover, we tried different methods of regression such as logarithmic, polynomial, lasso, stepwise and elasticnet but these methods did not outperform the linear regression.

**Final Model Selection:**

We were chosen between our own built CNN and Linear Regression models, but we decided to choose Linear Regression model. This model outperforms results for CNN, unless we increase number of epochs in CNN. Even though we get sharper pattern for predicted image than CNN, the intensity of images for Linear Regression is much better than for CNN. The predicted centers for test images can be retrieved from linearregression.txt file.

**Team responsibilities:**

Both team members had separate prediction methods and tested it. Then one team member was responsible for KNN and Decision Tree methods and another was responsible for CNN and Alexnet methods. The results and methods were compared by online chat.