CS215 - Assignment 3

Ramachandran S - 23b1052
Keyaan KR - 23b0977
Harith S - 23b1085

October 11, 2024

Contents

1 Finding optimal bandwidth

1.1.a

Lemma

Let $\hat{f}(x)$ be a histogram estimator with bandwidth h, where v_j represents the number of points in the j-th bin and n is the total number of points in the dataset. Then the integral of the square of the estimator is given by:

$$\int \hat{f}(x)^2 \, dx = \frac{1}{n^2 h} \sum_{j=1}^{m} v_j^2$$

Given:

$$\hat{f}(x) = \sum_{j=1}^{m} \frac{v_j}{nh} \cdot \mathbb{1}_{[x \in B_j]}$$

where $\mathbb{1}_{[x \in B_j]}$ is the indicator function, which equals 1 if x is in the j-th bin and 0 otherwise.

Proof:

Now, we square the histogram estimator:

$$\hat{f}(x)^2 = \left(\sum_{j=1}^{m} \frac{v_j}{nh} \cdot \mathbb{1}_{[x \in B_j]} \right)^2$$

$$\hat{f}(x)^2 = \sum_{j=1}^{m} \frac{v_j^2}{n^2 h^2} \cdot \mathbb{1}[x \in B_j] + 2 \cdot \sum j = 1^m \sum_{k \neq j} \frac{v_j v_k}{n^2 h^2} \cdot \mathbb{1}[x \in B_j] \cdot \mathbb{1}[x \in B_k]$$

We can observe that $\mathbb{1}[x \in B_j] \cdot \mathbb{1}[x \in B_k] = 0$ because x can only belong to either B_j or B_k, but not both, since the bins B_j and B_k are non-overlapping. Hence, the cross terms disappear. Therefore, the expression simplifies to:

$$\hat{f}(x)^2 = \sum_{j=1}^{m} \frac{v_j^2}{n^2 h^2} \cdot \mathbb{1}_{[x \in B_j]}$$

Now, integrate $\hat{f}(x)^2$ over the entire domain:

$$\int \hat{f}(x)^2 \, dx = \int \sum_{j=1}^{m} \frac{v_j^2}{n^2 h^2} \cdot \mathbb{1}_{[x \in B_j]} \, dx$$

$$\int \hat{f}(x)^2 \, dx = \sum_{j=1}^{m} \frac{v_j^2}{n^2 h^2} \int_{B_j} dx \qquad \text{as } \hat{f}(x)^2 \text{ is constant within each bin } B_j$$

The integral $\int_{B_j} dx$ over each bin is just the bandwidth of the bin, which is h. Substituting this back into the equation, we get:

$$\int \hat{f}(x)^2 \, dx = \sum_{j=1}^{m} \frac{v_j^2}{n^2 h^2} \cdot h$$

$$\int \hat{f}(x)^2 \, dx = \frac{1}{n^2 h} \sum_{j=1}^{m} v_j^2$$

1.1.b

Lemma

Let $\hat{f}_{(-i)}(X_i)$ be the histogram estimator after removing the i^{th} observation from the dataset. Then the sum of the estimator over all points is given by:

$$\sum_{i=1}^{n} \hat{f}(-i)(X_i) = \frac{1}{(n-1)h} \sum j = 1^m (v_j^2 - v_j)$$

Proof:

We know $\hat{f}(x) = \sum_{j=1}^{m} \frac{v_j}{nh} \cdot \mathbb{1}_{[x \in B_j]}$. Therefore, the estimator after removing the i^{th} observation is given by:

$$\hat{f}(-i)(X_i) = \sum j = 1^m \frac{v_j - \mathbb{1}[X_i = x_i]}{(n-1)h} \cdot \mathbb{1}[X_i \in B_j]$$

Thus, by summing over all i, we get:

$$\sum_{i=1}^{n} \hat{f}(-i)(X_i) = \sum i = 1^n \sum_{j=1}^{m} \frac{v_j - \mathbb{1}[X_i = x_i]}{(n-1)h} \cdot \mathbb{1}[X_i \in B_j]$$

$$\sum_{i=1}^{n} \hat{f}(-i)(X_i) = \frac{1}{(n-1)h} \cdot \sum j = 1^m \sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j]$$

The inner summation, $\sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j]$, can be simplified as:

$$\sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j] = \sum_{i=1}^{n} v_j \cdot \mathbb{1}[X_i \in B_j] - \sum i = 1^n \mathbb{1}[X_i = x_i] \cdot \mathbb{1}[X_i \in B_j]$$

$$\sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j] = v_j \cdot \sum_{i=1}^{n} \mathbb{1}[X_i \in B_j] - \sum i = 1^n \mathbb{1}[X_i = x_i] \cdot \mathbb{1}[X_i \in B_j]$$

The first summation, $v_j \cdot \sum_{i=1}^{n} \mathbb{1}[X_i \in B_j]$, is simply the count of points in bin B_j, which is v_j. The second summation, $\sum i = 1^n \mathbb{1}[X_i = x_i] \cdot \mathbb{1}[X_i \in B_j]$, is v_j since for each point in bin B_j, the indicator function is 1. Thus, the inner summation simplifies to:

$$\sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j] = v_j \cdot v_j - v_j$$

$$\sum_{i=1}^{n} (v_j - \mathbb{1}[X_i = x_i]) \cdot \mathbb{1}[X_i \in B_j] = v_j^2 - v_j$$

Therefore, we can express the sum of the estimator over all points as:

$$\sum_{i=1}^{n} \hat{f}(-i)(X_i) = \frac{1}{(n-1)h} \sum j = 1^m (v_j^2 - v_j)$$

1.2.a

Histogram was plotted with the number of bins equal to 10. The estimated probabilities $\hat{p_j}$ for all the bins can be found below:

Bin Number	Estimated Probability
1	0.00588235
2	0.38235294
3	0.32352941
4	0.05294118
5	0.11176471
6	0.10000000
7	0.00588235
8	0.00000000
9	0.01176471
10	0.00588235

Table 1: Estimated Probabilities for the 10 Bins

1.2.b

The histogram with 10 bins may likely be an underfit. When the bins are too wide, we observe that the histogram smooths over finer details, resulting in an underfit.

1.2.c

The cross-validation score formula was used to compute cross validation score for the bin width h, corresponding to 1 to 1000 bins. The resulting plot was saved in `crossvalidation.png`

1.2.d

Based on the above observation, the optimal value of bin width is 0.0833333 and the corresponding number of bins is 48.

1.2.e

The histogram corresponding to optimal bandwidth is saved in `optimalhistogram.png`. This histogram is different from the 10 bins histogram becuase this histogram captures finer details better and we get to know the estimated probabilities in every range in a better way.

2 Detecting Anomalous Transactions using KDE

The range of values for which the plot has been made is from $(-6, 6) \times (-6, 6)$.

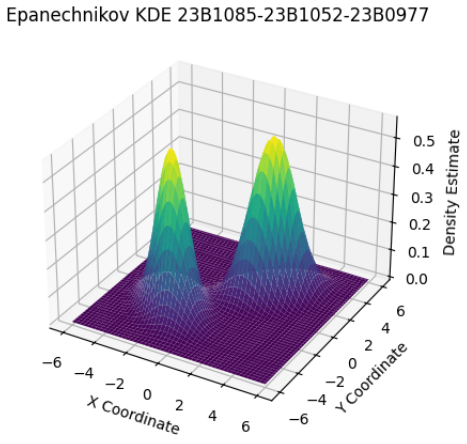

Figure 1: Transaction Distribution

The graph has a bimodal distribution, having two peaks at around **(2,4)** and **(-2,-3)**. If transactions fall within regions with have a probability which is lesser than a certain threshold, we can flag it as suspicious.

3 Simple Linear Regression Model

3.1 SubPart-1

In a simple linear regression model, we have the equation:

$$y = \beta_0 + \beta_1 x + \epsilon$$

where y is the dependent variable, x is the independent variable, β_0 is the intercept, β_1 is the slope, and ϵ is the error term. The least squares estimates for β_0 and β_1 are given by:

$$\hat{\beta}1 = \frac{\sum i = 1^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

where $\bar{x}$ and $\bar{y}$ are the sample means of x and y, respectively.

Proof that $(\bar{x}, \bar{y})$ lies on the Regression Line

The equation of the least squares regression line is:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

Substituting $x = \bar{x}$, the mean of the x-values, into this equation gives:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \bar{x}$$

Substituting the expression for $\hat{\beta}_0$ into the equation:

$$\hat{y} = (\bar{y} - \hat{\beta}_1 \bar{x}) + \hat{\beta}_1 \bar{x}$$

Simplifying this, we get:

$$\hat{y} = \bar{y}$$

Thus, when $x = \bar{x}$, we have $\hat{y} = \bar{y}$. Therefore, the point $(\bar{x}, \bar{y})$ lies on the least squares regression line.

3.2 SubPart-2

Original Model

The original simple linear regression model is:

$$Y = \beta_0 + \beta_1 x + \epsilon$$

where Y is the dependent variable, x is the independent variable, and ϵ is the error term. The least squares estimates of β_0 and β_1 are:

$$\hat{\beta}1 = \frac{\sum i = 1^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

New Model

Consider the new regressor $z = x - \bar{x}$, where z is the centered version of x. The new model becomes:

$$Y = \beta_0^* + \beta_1^* z + \epsilon$$

The new model is expressed in terms of z rather than x. Since $z = x - \bar{x}$, the least squares estimates of β_0^* and β_1^* will be obtained in the same way, but with z replacing x.

Step 1: Estimating β_1^*

Computing β_1^* in the same way, we get

$$\hat{\beta}1^* = \frac{\sum i = 1^n (z_i - \bar{z})(y_i - \bar{y})}{\sum_{i=1}^n (z_i - \bar{z})^2}$$

Since $z = x - \bar{x}$, it's mean becomes $\bar{x} - \bar{x} = 0$. Substituting $\bar{z} = 0$, we get:

$$\hat{\beta}1^* = \frac{\sum i = 1^n (z_i)(y_i - \bar{y})}{\sum_{i=1}^n z_i^2}$$

Substituting $z_i = x_i - \bar{x}$ gives:

$$\hat{\beta}1^* = \frac{\sum i = 1^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

Thus, we have:

$$\hat{\beta}_1^* = \hat{\beta}_1$$

Step 2: Estimating β_0^*

The estimate for β_0^* is given by:

$$\hat{\beta}_0^* = \bar{y} - \hat{\beta}_1^* \bar{z}$$

Since $\bar{z} = 0$ (because $z = x - \bar{x}$ and its mean is zero), we have:

$$\hat{\beta}_0^* = \bar{y}$$

Relationship Between the Least Square Estimates

- In the original model, the least squares estimates are:

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

$$\hat{\beta}_1 = \hat{\beta}_1$$

- In the new model, the least squares estimates are:

$$\hat{\beta}_0^* = \bar{y}$$

$$\hat{\beta}_1^* = \hat{\beta}_1$$

How Are the Models Different?

1. **Intercept**: In the original model, $\hat{\beta}_0$ accounts for both the mean of y and the mean of x, whereas in the centered model, the intercept $\hat{\beta}_0^*$ is simply the mean of y.
2. **Interpretation**: The new model, where $z = x - \bar{x}$, is often simpler to interpret since the intercept $\hat{\beta}_0^*$ represents the mean of y, i.e., the value of y when x is at its mean, making the intercept more meaningful. The intercept in the original model gives the predicted value of y when x=0, but this can be less meaningful if x=0 is far from the observed range of the data.

3.3 SubPart-3

3.3.a

The code file for performing **OLS** from scratch has been uploaded. It receives data to train the model from `train.csv` and test data is taken from `test.csv`. The model is then trained and the predictions for the test data is written onto `3_predictions.csv`. The parameters learnt are saved in `3_weights.pkl`.

3.3.b

For determining the degree of polynomial that gives us the best fit, we will use MSE to determine which degree returns the least MSE. For the test data given, the best fit is obtained for degree = 6. The degree that gives underfit is 3 and degree = 16 gives overfit. A plot displaying all the three fits is shown below

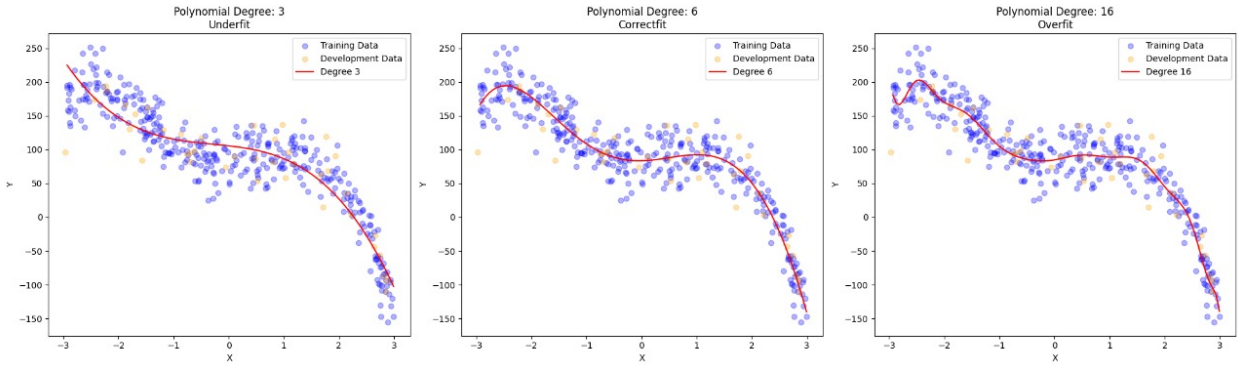

Figure 2: Comparison of underfit, best fit, and overfit models

3.3.c

Degree	R^2
3	0.7891
6	0.8740
16	0.8409

Table 2: Degrees of polynomial and their corresponding R^2 values

From the above table, you can infer that degree = 6 gives the highest Coefficient of Determination.

4 Non-parametric regression

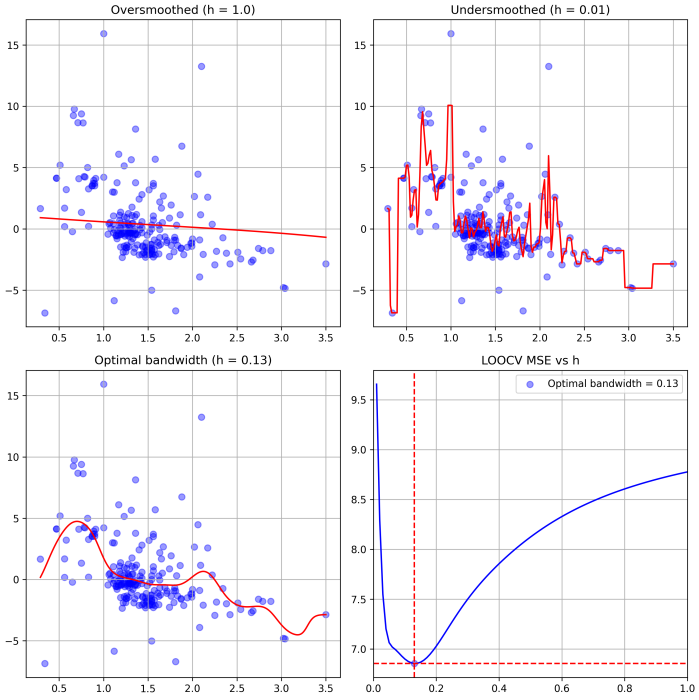

Figure 3: Gaussian Kernel Regression

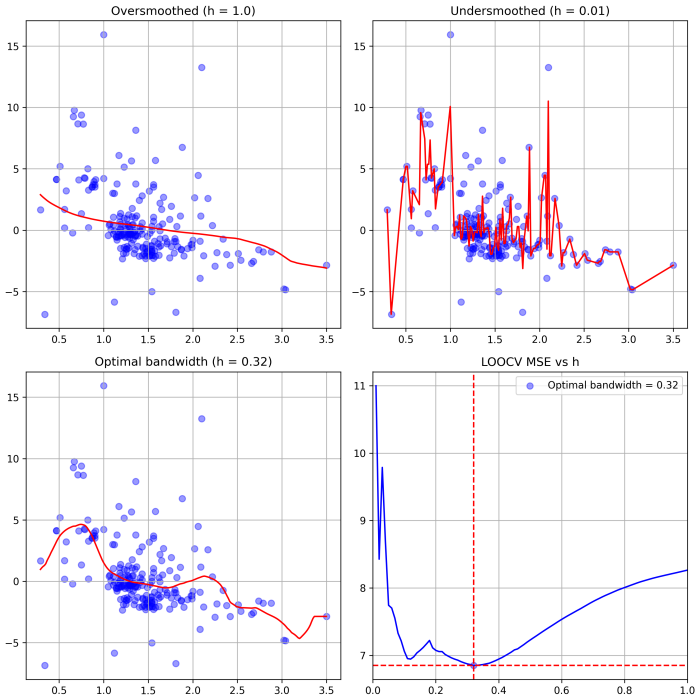

Figure 4: Epanechnikov Kernel Regression

The bandwidth corresponding to the minimum risk is as follows:

- **Epanechnikov Kernel Regression**: 0.32
- **Gausian Kernel Regression**: 0.13

Similarities:

- The optimal plots have roughly the same shape. Corresponding to a point, the most likely y as a result is roughly the same point, showing consistency with the fits.
- Both functions follow properties that Kernel functions have: Both are symmetric, tend to zero as input tends to infinity, and both return probability density functions.

Differences:

- The final optimal bandwidth of both differ significantly.
- Epanechnikov gives a much more sharp curve compared to the Gaussian.
- The loss function also is smooth for Gaussian as opposed it being sharp in Epanechnikov.
- The number of modes also can differ in case there are values occuring frequently close together but separated slightly. Gaussian approximates this and returns a unimodal prediction as opposed to Epanechnikov when data sampled from a Normal distribution is used.
- The jagged appearance of Epanechnikov is due to the limited smoothness of the quadratic function, that is after a point, the derivative vanishes.
- The support for Gaussian extends for all real numbers whereas for Epanechnikov, it is only between -1 and 1.
- Gaussian can also be more computationally expensive as calculating a quadratic which has finite support is much simpler than doing so across all real numbers.

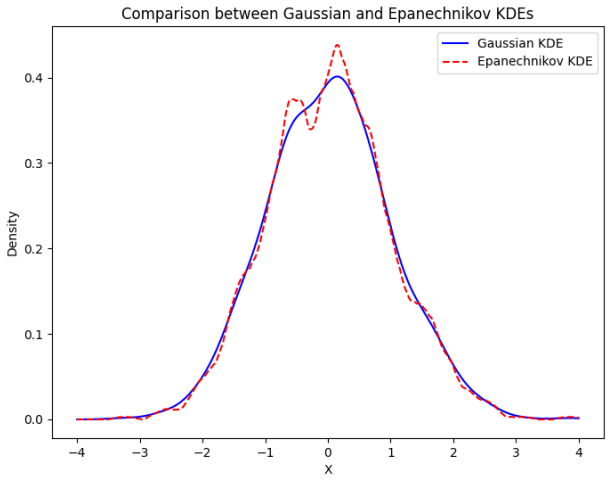

Figure 5: Difference between Gaussian and Epanechnikov for a normal distribution

5 Multivariate Insights Unlocked

5.1 Dependent variables

After the data is read into df and dftest, the range is yield values is checked. This gives an idea of how the final prediction should look like. When the correlation matrix is plotted, we see that a bunch of features

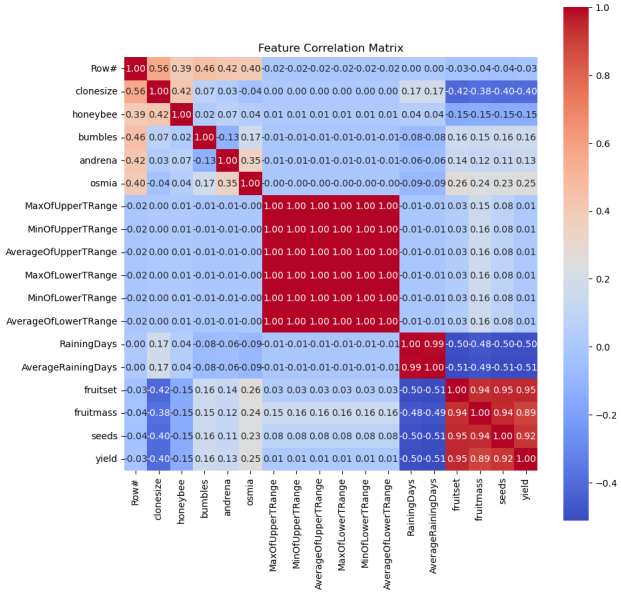

Figure 6: Correlation between target and features

are correlated, and some in fact are linearly dependent. Although this doesn't affect the predictions, it's much more efficient to have a single independent feature, because of lower number of features to calculate. This again is verified by checking the multi-col linearity using variance inflation.

	feature	VIF
0	Row#	4.208068e+00
1	clonesize	2.402712e+00
2	honeybee	1.331307e+00
3	bumbles	1.932668e+00
4	andrena	1.819964e+00
5	osmia	1.431468e+00
6	MaxOfUpperTRange	3.373617e+06
7	MinOfUpperTRange	9.973142e+05
8	AverageOfUpperTRange	1.025869e+05
9	MaxOfLowerTRange	5.356488e+03
10	MinOfLowerTRange	9.263639e+06
11	AverageOfLowerTRange	1.521515e+03
12	RainingDays	6.139146e+01
13	AverageRainingDays	6.240067e+01
14	fruitset	1.958773e+01
15	fruitmass	1.351198e+01
16	seeds	1.365369e+01

Figure 7: Multi-Collinearity

For analysis after this, since it would involve changing the data frame, we create copies of the original data frame.

5.2 Feature reduction

Features relating to Temperature are highly correlated. So, we'll keep one of them, and check for VIF. And, AvgRainingDays and RainingDays have a linear relationship. Since Avg has a higher magnitude of correlation, we'll keep it. This feature reduction is reflected on top as well, although changes here have been done to a copy of the data frame. Moreover, Row# at first sight seems to be a ID variable which is confirmed by the scatter plot. Seeing the huge variance and no common theme, it can be inferred that this was an index variable.
We also take help of Feature Importance using Random Forest and Mutual Information to infer which would be the best feature to keep around(as keeping around noisy features can confuse the model) and which to reduce.

5.3 Checking for outliers

Outliers can affect the performance of the Linear Regressor. That is why it is important to check for outliers. For this, we scattar plot with respect to the remaining features. On checking however, only *honeybee* seems to have outliers, and changing it didn't cause much difference to the prediction.

5.4 Normalising the data

since we are raising features to some exponent, it is very possible that the values get out of control and start causing huge changes for very minute changes. So, it is important to normalise the data. For this, we use the normalise function.

5.5 OLS

OLS is based on the following equation:

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

which minimizes the error between the target and the prediction. For adding polynomial regression with respect to each feature, we simply add another feature which is the current feature raised to the exponent.

5.6 Ridge regulation

We also have to leaf with overfitting. For that we use Ridge regulation. The implementation is very similar to that of the OLS, but it penalises very large co-efficients.

5.7 Assigning degree based on the correlation

So far we have been assigning the same degree to everything. The degree represents the level of dependence the target has. So, it will be better to assign a greater degree to a feature which is highly correlated as opposed to one which isn't.

5.8 Kernel Regression

Kernels are much better used as classificators rather than regressors. This is because of the immense time it takes to train a particular model.