Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

## 3.1 Before Training

### 3.1.1 Data Inspection

**QUESTION 1:** Plot a heatmap of Pearson correlation matrix of dataset columns.

**Bike dataset**

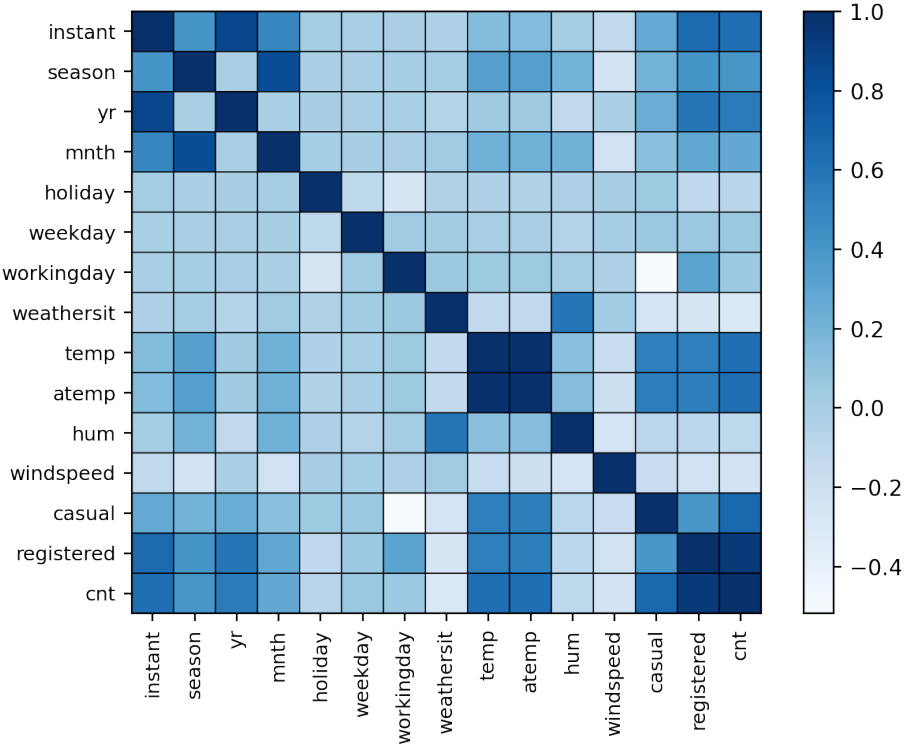



Figure 1: Grid matrix of bike dataset

For target *casual*: feature *atemp* has the highest absolute correlation value of 0.543864

For target *registered*: feature *instant* has the highest absolute correlation value of 0.659623

For target *cnt*: feature *atemp* has the highest absolute correlation value of 0.631066

It implies that the features *atemp* is most correlated to the targets.

Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

**Suicide dataset**

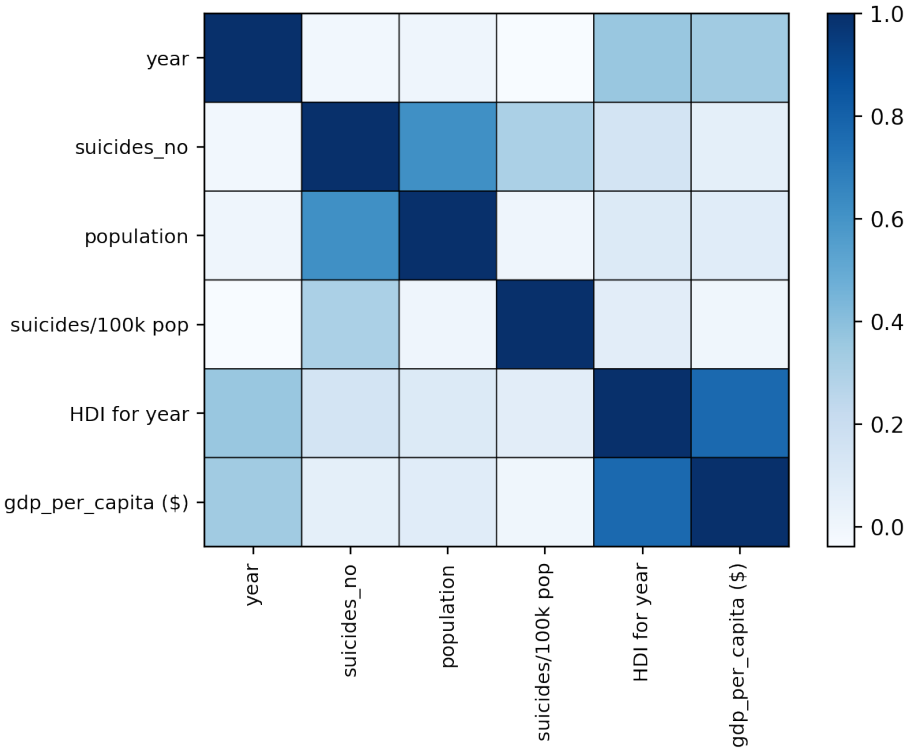



Figure 2: Grid matrix of suicide dataset

For target *suicides_no*: feature *population* has the highest absolute correlation value of 0.616162

For target *suicides*/100*kpop*: feature $HDIforyear$ has the highest absolute correlation value of 0.074279

For the suicide dataset, the highest correlation was reported for the feature population = 0.616162 for suicides-no and for suicides/100 k population, highest correlation was found for the feature suicides-no = 0.306604.

This shows that the numbers influence correlation to a great extent. The population influences it for the number of suicides, while the number of suicdes reports a high correlation for the number of suicides/100k population, which is natural, because a high suicide would report a higher percentage as well.

Thus, population, number of suicides and number of suicides per 100k population are closely correlated.

Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

## Video dataset

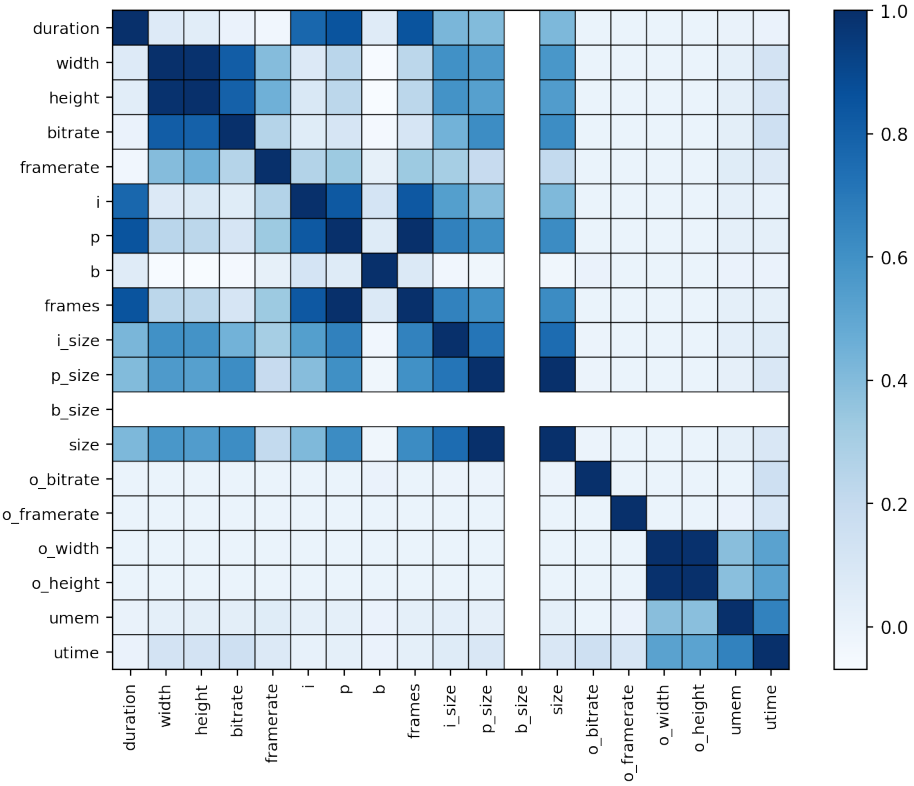



Figure 3: Grid matrix of video dataset

For target *utime*: feature *o_width* has the highest absolute correlation of 0.523388

This makes sense as the larger the output width of the video is, the more time transcoding would take.

**QUESTION 2:** Plot the histogram of numerical features.

### Bike Dataset

Numerical features for bike dataset include: *temp*, *atemp*, *hum*, *windspeed*.

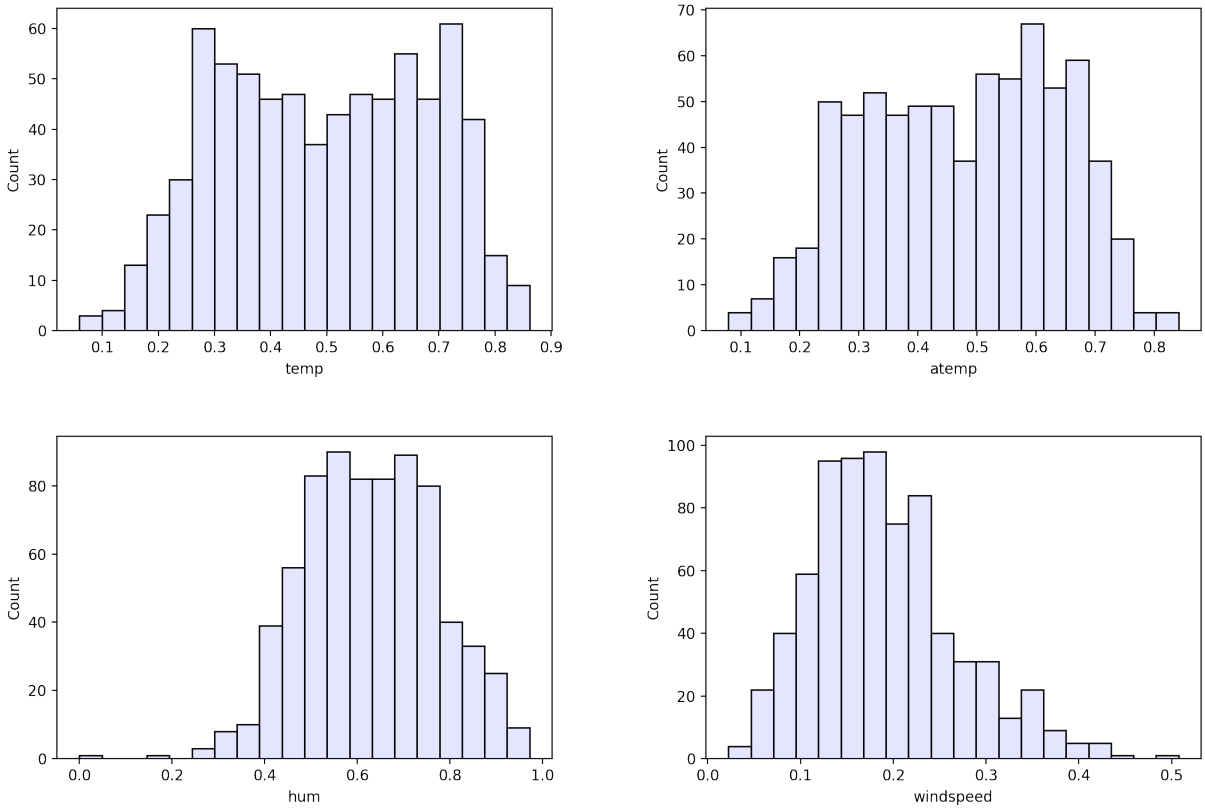



Figure 4: Histogram of numerical features in bike dataset

We can see the histograms of numerical features in the bike dataset are all more-or-less normal distributed without skewness.

Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

### Suicide Dataset

Numerical features for suicide dataset include: *population*, $HDI\ for\ year$, *gdp_for_year* ($), *gdp_per_capita* ($).

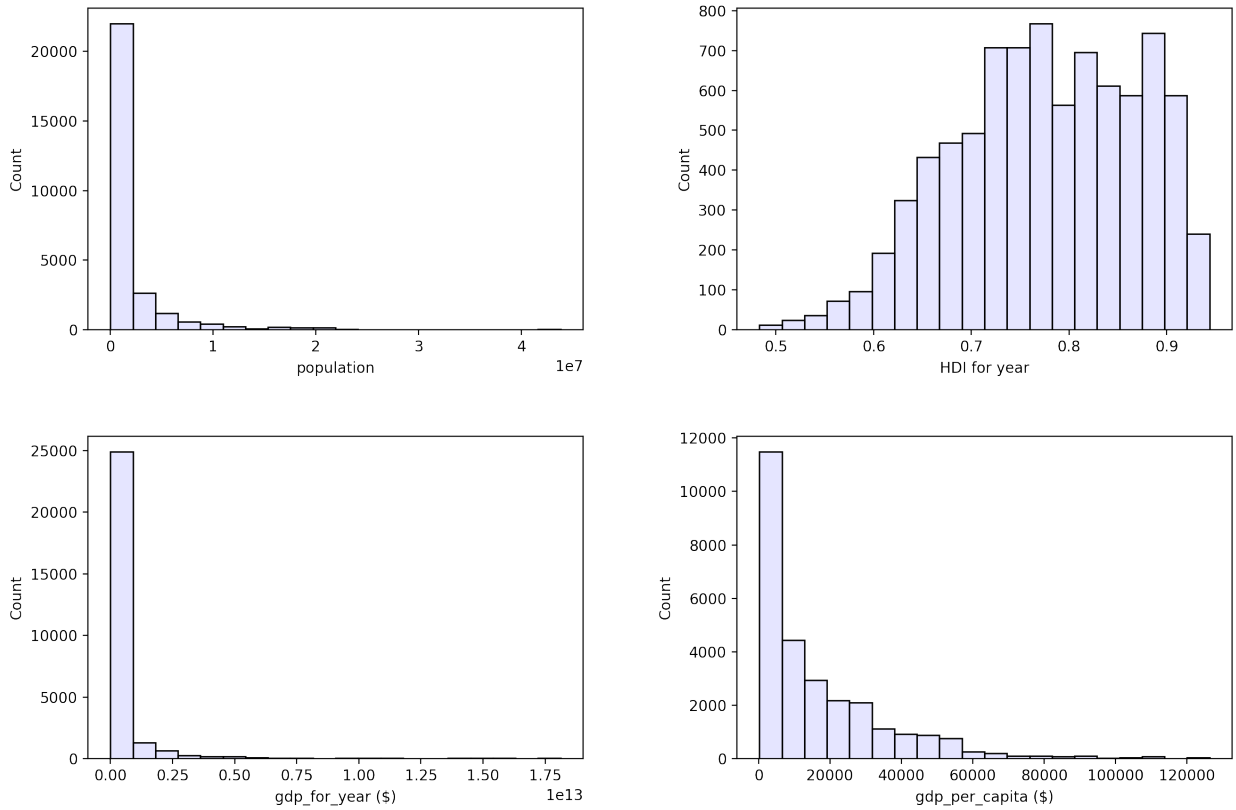



Figure 5: Histogram of numerical features in suicide dataset

We notice the feature *population*, *gdp_for_year* ($), and *gdp_per_capita* ($) has high skewness, so we calculate the log value of these features in the preprocessing to make them more or less similar to normal distribution.

Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

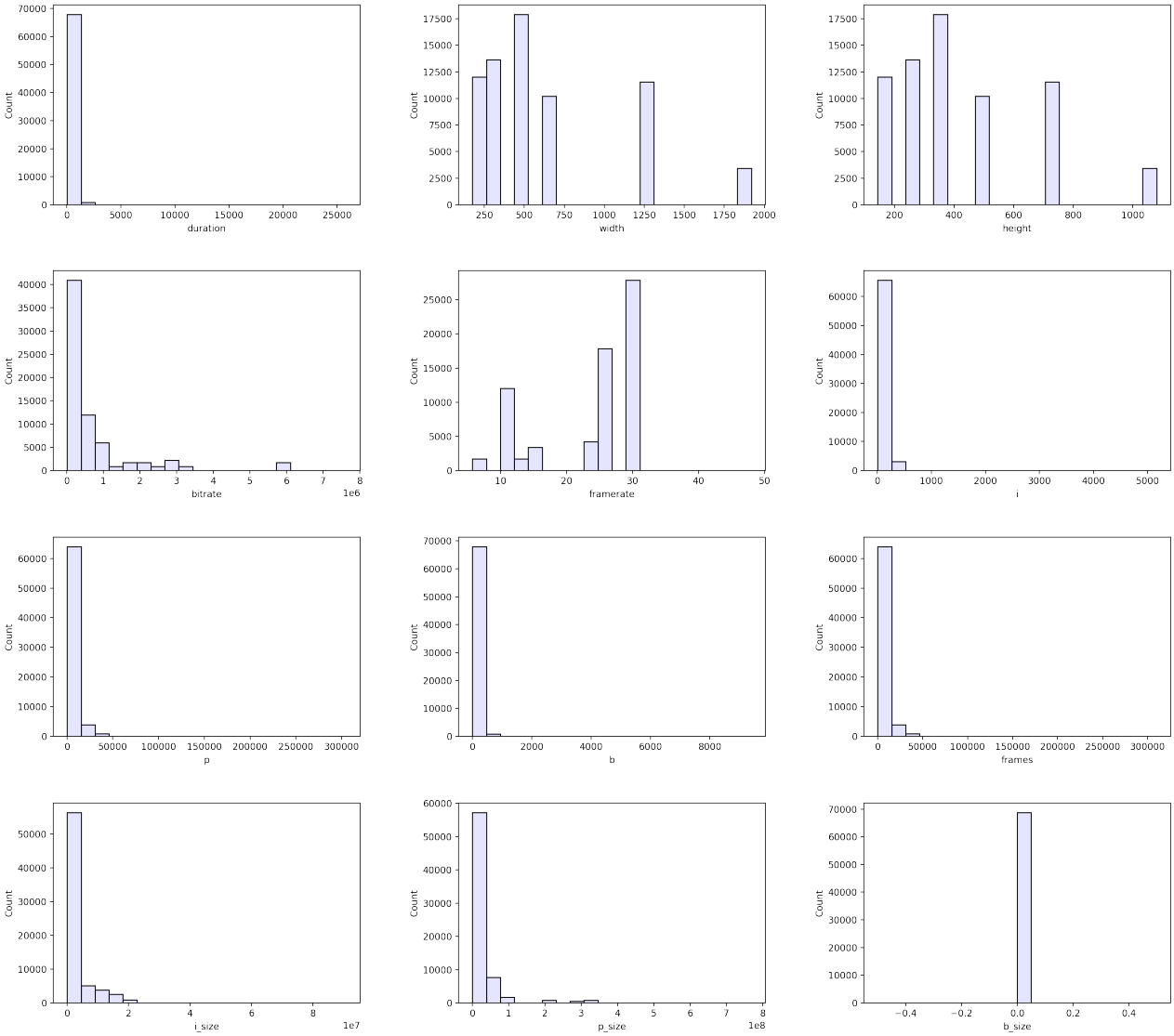
# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

## Video Dataset

Numerical features for video dataset include: *duration*, *width*, *height*, *bitrate*, $framerate$, *i*, *p*, *b*, $frames$, *i_size*, *p_size*, *b_size*, *size*, *o_bitrate*, $o\_framerate$, *o_width*, *o_height*.



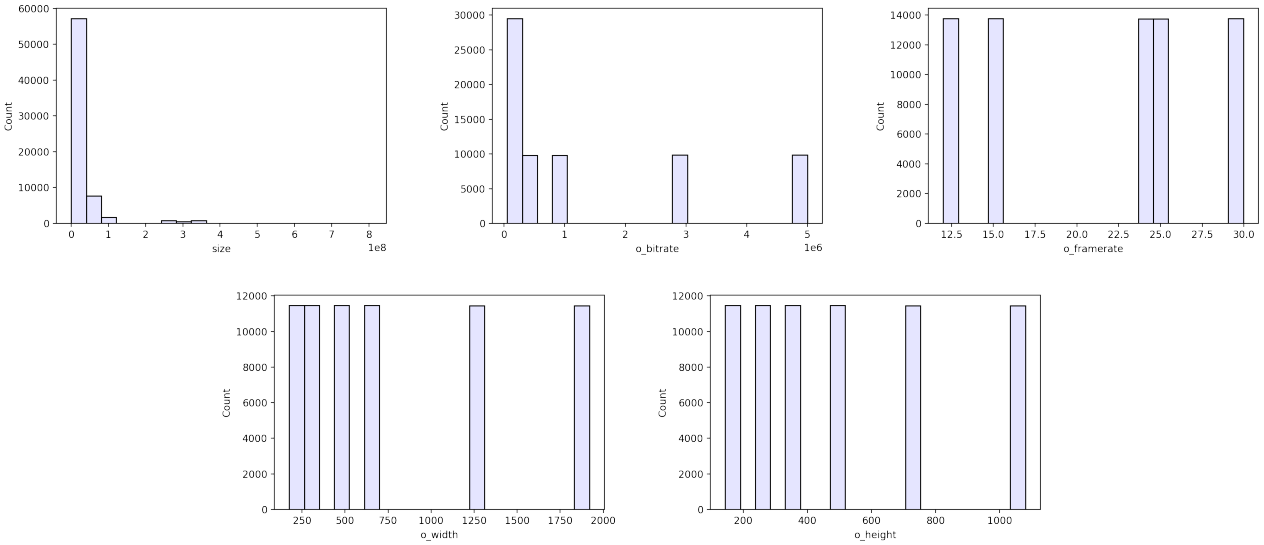



Figure 6: Histogram of numerical features in video dataset

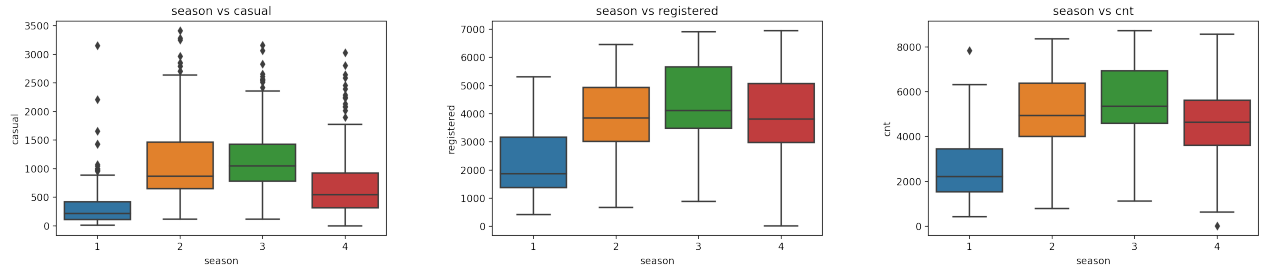
From the histograms in Fig. 6, we can see that many of the numerical features in video dataset are very sparse in x-axis, indicating that they are more similar to categorical features than numerical features.

For features with high skewness, such as *bitrate*, *i_size*, *p_size*, we again calculate their log values to have a normal distribution.

**QUESTION 3:** Inspect the box plot of categorical features vs target variable.

### Bike Dataset

Categorical features for bike dataset include: *season*, *yr*, *mnth*, *holiday*, *weekday*, *workingday*, *weathersit*.

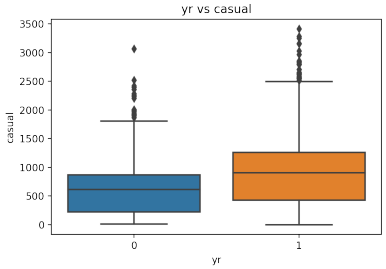


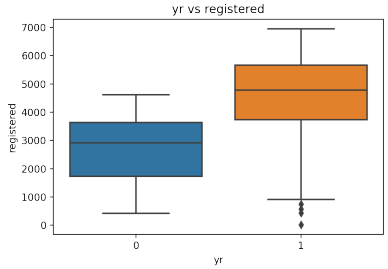
Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

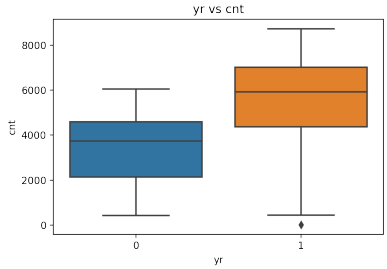
# Project 4 Report

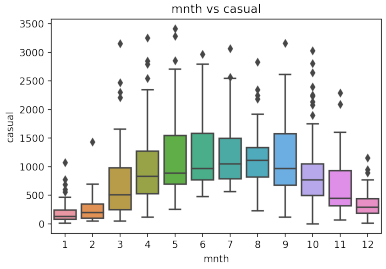
## Regression Analysis

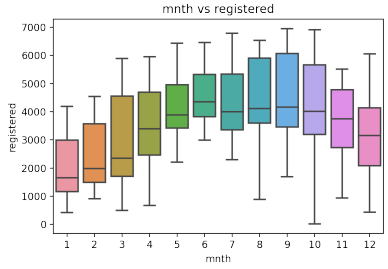
ECE 219
Data Mining
March 21, 2021

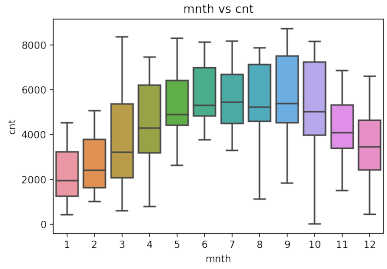


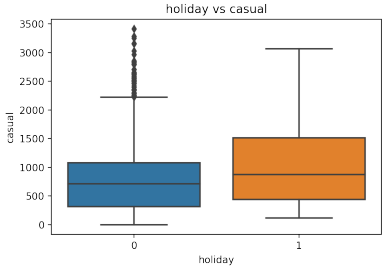



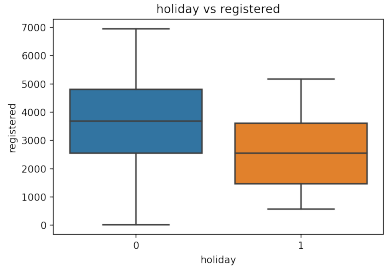



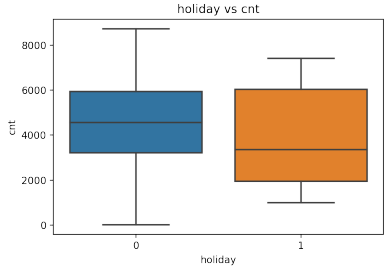



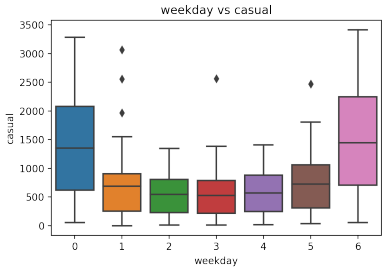



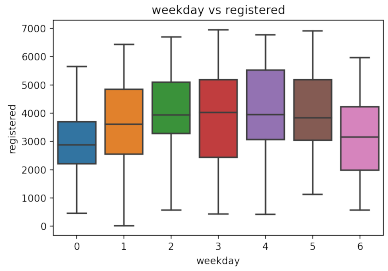



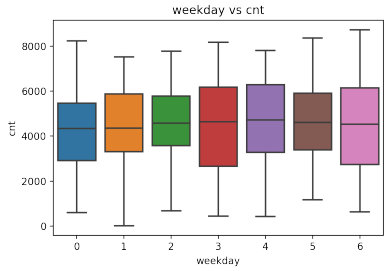



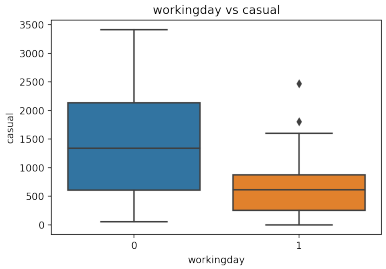



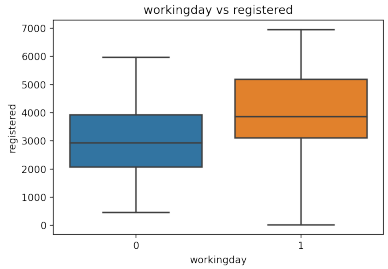



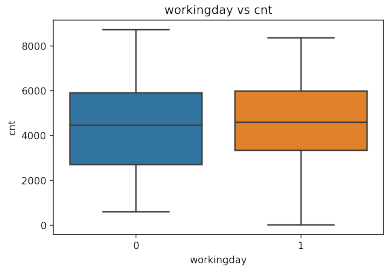



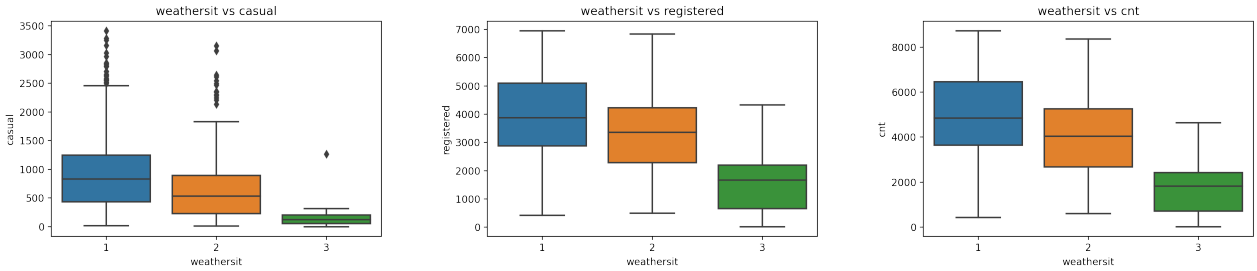
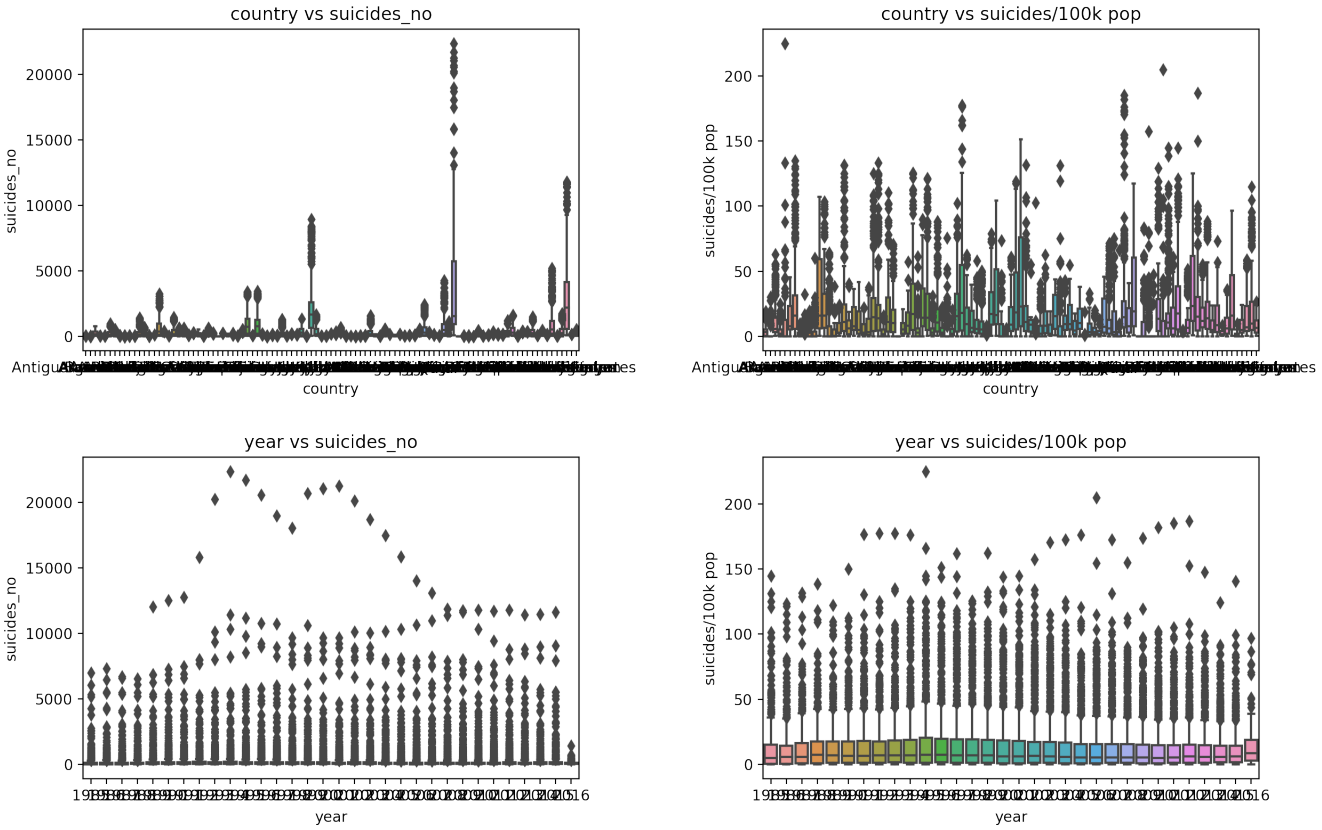



Figure 7: Box plots of categorical features vs target variables for bike dataset

For the bike dataset, we notice that *weekday* and *workingday* vs *cnt* shows equal distributions among various feature values, indicating that the count of total rental bikes is not influenced by the day of the week.

### Suicide Dataset

Categorical features for suicide dataset include: *country*, *year*, *sex*, *age*, *generation*.



Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

**Regression Analysis**

ECE 219
Data Mining
March 21, 2021

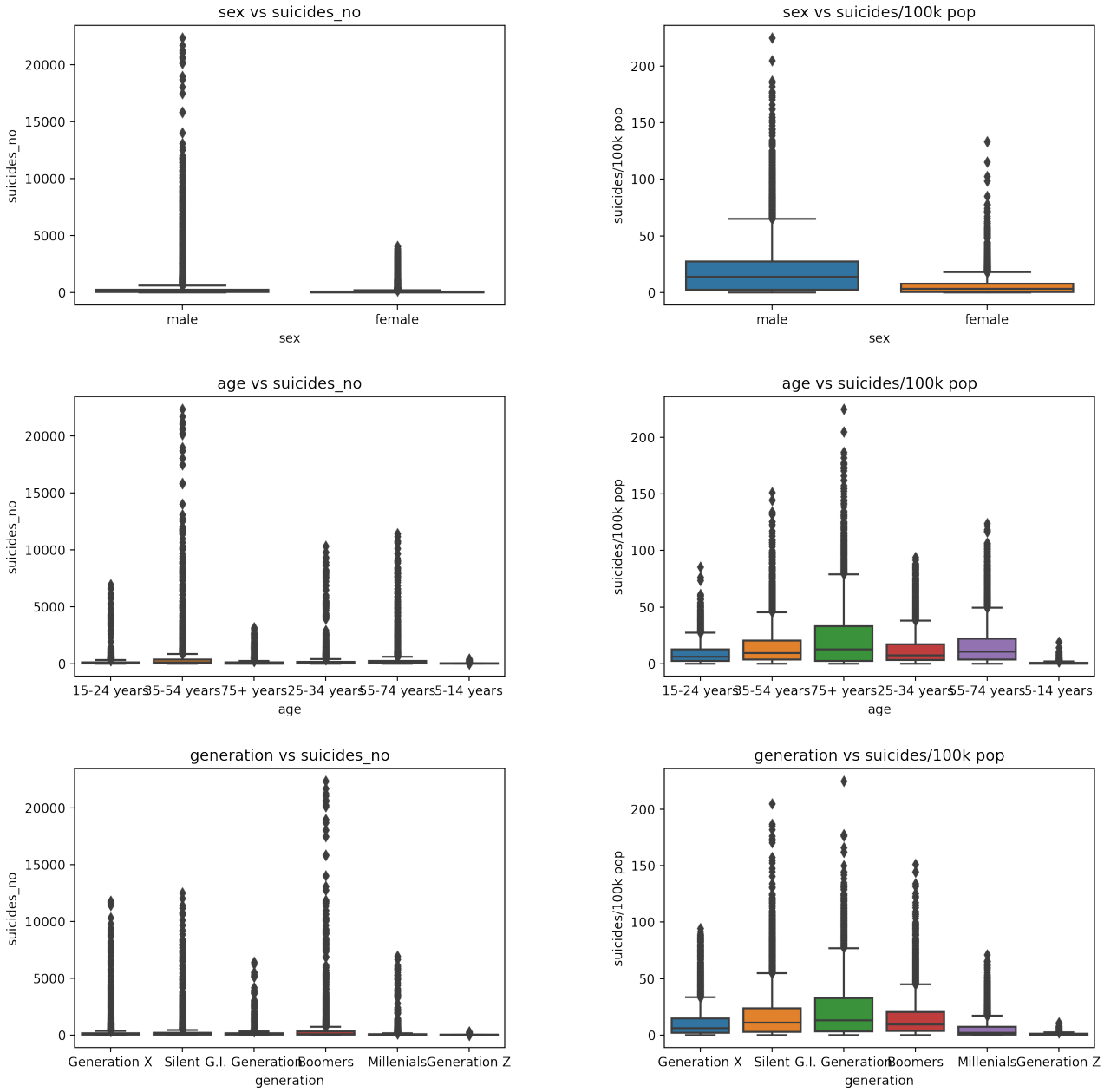



Figure 8: Box plots of categorical features vs target variables for suicide dataset

For the suicide dataset, the box plot gives us some intuition that the feature *sex*, *age*, *generation* are very closely related with suicide number.

### Video Dataset

Categorical features for video dataset include: *codec*, *o_codec*.

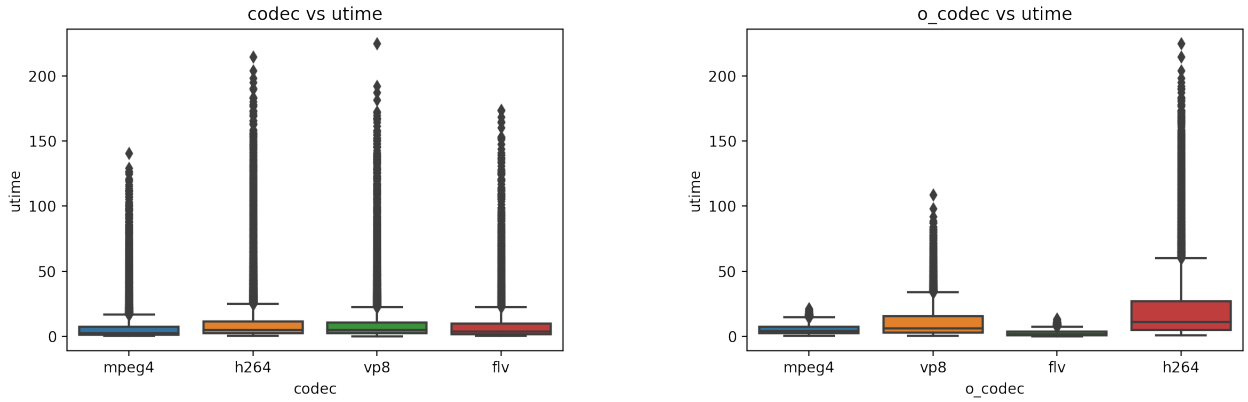



Figure 9: Box plots of categorical features vs target variables for video dataset

From the box plots for video dataset, we notice that the feature *codec* doesn't influence *utime* much while *o_codec* shows some large influence.

**QUESTION 4:** For bike sharing dataset, plot the count number per day for a few months. Can you identify any repeating patterns in every month?

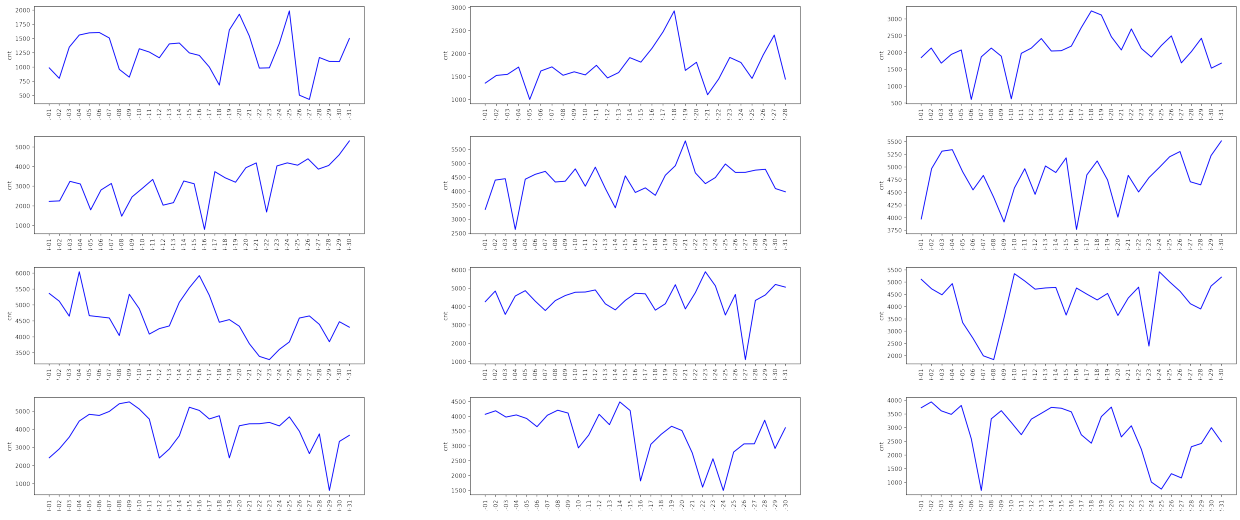


Figure 10: *cnt* vs data for 12 months

Most months show a decrease in the initial days of the month, but they generally show an increase during the middle of the month. Finally, it's a 50:50 trend in the end whether it increases or decreases.

Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

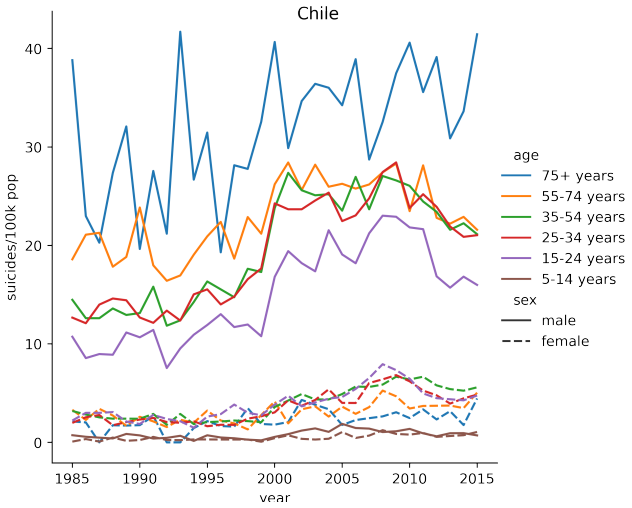
# Project 4 Report

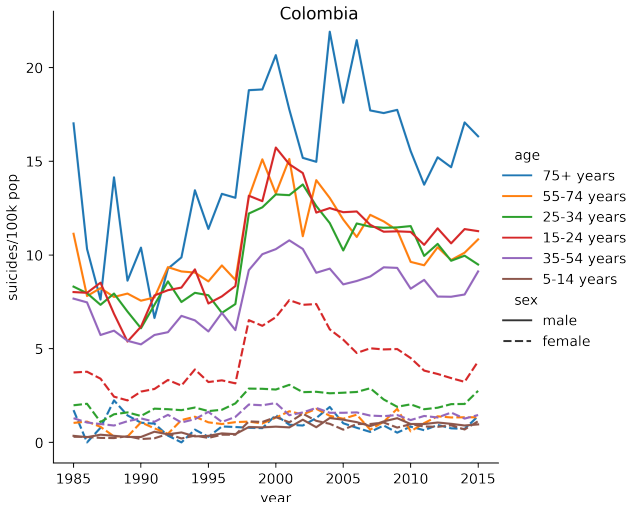
**Regression Analysis**

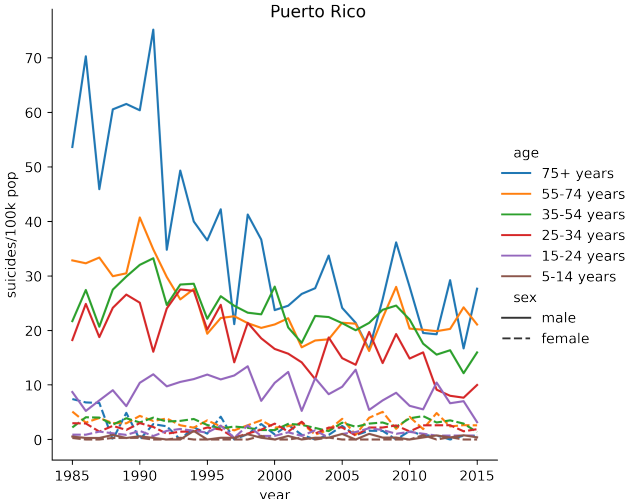
ECE 219
Data Mining
March 21, 2021

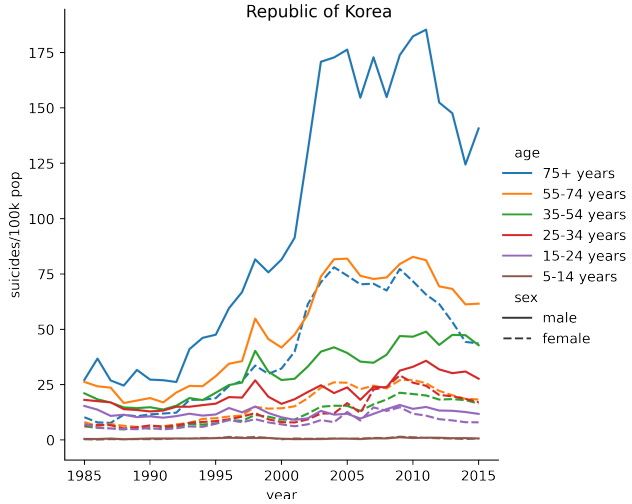
**QUESTION 5:** For the suicide rate dataset, pick the top 10 countries that have the longest time-span of records (in terms of years). Plot the suicide rate against time for different age groups and gender.

The ten countries that we selected to plot are: Chile, Columbia, Puerto Rico, Republic of Korea, Brazil, Mexico, Netherlands, Austria, Iceland, Mauritius.



Qiong Hu (405065032)
Daniel Ahn (504817439)
Riyya Hari Iyer (305427411)
Juan Piao (304959335)

# Project 4 Report

## Regression Analysis

ECE 219
Data Mining
March 21, 2021


Figure 11: Suicide rate against time for different groups in ten countries

It has been observed that while most countries report an increasing trend over the past few years, only Puerto Rico and Austria seem to show a gradual decrease over time while Mexico and Korea signal an increase over the years.

Also, by comparing different age groups and gender, we notice that male tend to have a larger suicide rate than female, and the age group of "75+ years" almost always have the highest suicide rate in all the countries and years.

**QUESTION 6:** For video transcoding time dataset, plot the distribution of video transcoding times.

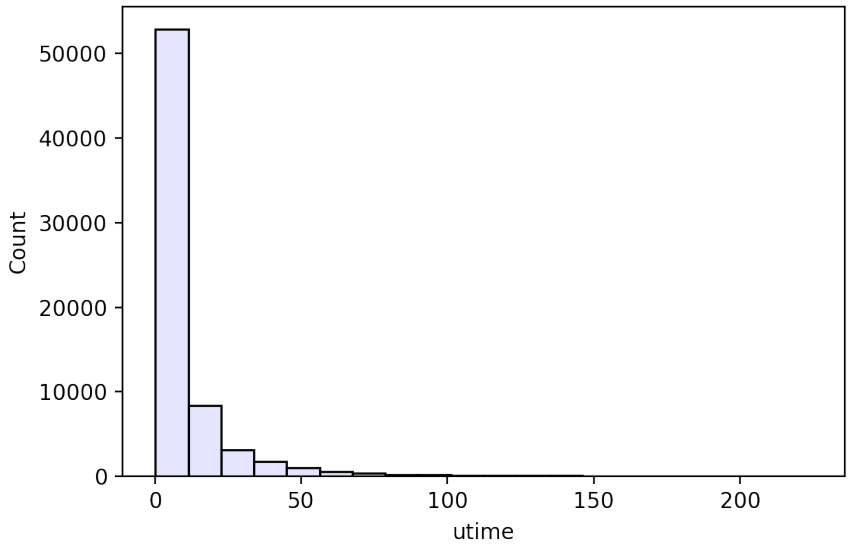



Figure 12: Distribution of video transcoding times

Mean Transcoding Time: 9.99635 seconds

Median Transcoding Time: 4.408 seconds

We observe that the count decreases as the *utime* increases, suggesting that the distribution of *utime* have a high skewness which requires preprocessing before training. Similar to histogram of features from Quetion 2, we also use the log value to replace the original value during training, which would generate a more-or-less normal distribution.

### 3.1.2 Handling Categorical Features

**QUESTION 7:**

The main difference between the scalar encoding and one-hot encoding for categorical features is whether or not the encoded numbers have numerical meanings.

The trade-off between the two methods are as follows:

The one-hot encoding introduces large sparsity to the feature space, as well as increasing the dimensionality.

Assume we perform linear regression, one-hot encoding would discard the potential relationship within each category, allowing more flexibility for the trained weight parameter to be different among values in the same category.

For example, when encoding time stamps such as $\{Mon, ..., Sun\}$, the scalar encoding would assume that all the choices in the category are equally distributed, which would not necessarily be held strongly if we perform the one-hot encoding.

In our experiments, with main concern about computational complexity, we mainly implemented scalar encoding for categorical features in all three datasets.

**Bike Dataset**

Categorical features: values after scalar encoding
season: [1, 2, 3, 4]
yr: [0, 1]
mnth: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
weekday: [0, 1, 2, 3, 4, 5, 6]
workingday: [0, 1]
weathersit: [1, 2, 3]

**Suicide Dataset**

Categorical features: values after scalar encoding
country: [0, 1, 2, ..., 99, 100]
year: [1985, 1986, 1987, ..., 2016]
sex: [0, 1]
age: [0, 1, 2, 3, 4, 5]
generation: [0, 1, 2, 3, 4, 5]
continent: [0, 1, 2, 3, 4, 5]

**Video Dataset**

Categorical features: values after scalar encoding
codec: [0, 1, 2, 3]
o_codec: [0, 1, 2, 3]

### 3.1.3 Standardization

**QUESTION 8:** Standardize feature columns and prepare them for training.

We applied the function "StandardScaler" from the library "sklearn.preprocessing" to standardize all the feature columns to zero mean and unit variance.

**Bike Dataset**

Table 1: Mean and variance of features in bike dataset

| feature name | mean | variance |
|---|---|---|
| season | 2.4966 | 1.2322 |
| yr | 0.5007 | 0.2500 |
| mnth | 6.5198 | 11.8994 |
| holiday | 0.0287 | 0.0279 |
| weekday | 2.9973 | 4.0137 |
| workingday | 0.6840 | 0.2161 |
| weathersit | 1.3953 | 0.2965 |
| temp | 0.4954 | 0.0334 |
| atemp | 0.4744 | 0.0265 |
| hum | 0.6279 | 0.0203 |
| windspeed | 0.1905 | 0.0060 |

We notice that the largest variance (11.8994 for *mnth*) and the smallest variance (0.0060 for *windspeed*) have a magnitude difference of $10^3$. Also, since most of the feature distribution follows normal distribution in bike dataset, it is reasonable to do a standardization on all features.

**Suicide Dataset**

Table 2: Mean and variance of features in suicide dataset

| feature name | mean | variance |
|---|---|---|
| country | 49.2753 | 862.715 |
| year | 2001.26 | 71.7223 |
| sex | 0.5 | 0.25 |
| age | 2.4994 | 2.9120 |
| generation | 2.7019 | 3.2754 |
| population | 1.84e6 | 1.53e13 |
| HDI for year | 0.7766 | 0.0087 |
| gdp_for_year ($) | 4.456e11 | 2.113e24 |
| gdp_per_capita ($) | 1.687e4 | 3.567e8 |

We notice that the largest variance (2.113e24 for $gdp\_for\_year$) and the smallest variance (0.0087 fir $HDI\ for\ year$) have a magnitude difference of $10^{27}$, it is necessary to do a standardization to avoid the feature with large variance mis-identified as important features by training model.

**Video Dataset**

Table 3: Mean and variance of features in video dataset

| feature name | mean | variance | feature name | mean | variance |
|---|---|---|---|---|---|
| codec | 1.6098 | 0.9715 | b | 9.1479 | 8559.12 |
| o_codec | 1.5046 | 1.2506 | frames | 6641.71 | 3.786e7 |
| duration | 286.414 | 82515.8 | i_size | 2.849e6 | 1.87e13 |
| width | 624.934 | 214522 | p_size | 2.18e7 | 2.598e15 |
| height | 412.572 | 57895.0 | size | 2.502e7 | 2.932e15 |
| bitrate | 6.937e5 | 1.200e12 | o_bitrate | 1.395e6 | 3.060e12 |
| framerate | 23.2413 | 52.1977 | o_framerate | 21.1909 | 44.4710 |
| i | 100.869 | 7184.97 | o_width | 802.337 | 3.720e5 |
| p | 6531.69 | 3.692e7 | o_height | 503.826 | 9.984e4 |

We notice that the largest variance (2.932e15 for *size*) and the smallest variance (0.9715 for *codec*) have a magnitude difference of more than $10^{15}$. With the same reason with suicide dataset, it is necessary to perform standardization on video dataset.

### 3.1.4 Feature Selection

**QUESTION 9:** Use mutual information and F score to select most important features.

**Bike Dataset**

Table 4: Mutual info and F score in bike dataset

| feature name | mutual info | feature name | F score |
|---|---|---|---|
| atemp | 0.4649 | atemp | 482.45 |
| temp | 0.3894 | temp | 473.47 |
| mnth | 0.3758 | yr | 344.89 |
| yr | 0.2776 | season | 143.97 |
| season | 0.2171 | weathersit | 70.729 |
| weathersit | 0.0656 | mnth | 62.005 |
| windspeed | 0.0555 | windspeed | 42.438 |
| hum | 0.0462 | hum | 7.4619 |
| weekday | 0.0446 | holiday | 3.4214 |
| workingday | 0.0235 | weekday | 3.3311 |
| holiday | 0.0110 | workingday | 2.7367 |

In Table 4, we sorted the features in the decreasing order of mutual information and F score, respectively. Therefore, for those features that are at the top for both measurements, we can be in some way confident that they are important features.

Therefore, intuitively, we consider *atemp*, *temp*, *yr* as important features for the bike dataset.

**Suicide Dataset**

Table 5: Mutual info and F score in suicide dataset

| feature name | mutual info | feature name | F score |
|---|---|---|---|
| population | 0.9198 | sex | 4729.94 |
| country | 0.5374 | gdp_for_year | 1761.65 |
| age | 0.2767 | population | 1625.22 |
| gdp_for_year | 0.2482 | generation | 339.449 |
| gdp_per_capita | 0.1908 | country | 250.275 |
| generation | 0.1529 | HDI for year | 162.623 |
| HDI for year | 0.1352 | gdp_per_capita | 97.2135 |
| sex | 0.1303 | age | 50.3255 |
| year | 0.0 | year | 18.9260 |

Similar to Bike dataset, we sort the features in the decreasing order of mutual information and F

score, but here we didn't see much collision between the order of two lists.

We can see *population* might be an important feature, and we can be sure that *year* is not important as it has zero mutual information with the target label and has the lowest F score.

**Video Dataset**

Table 6: Mutual info and F score in video dataset

| feature name | mutual info | feature name | F score |
|---|---|---|---|
| bitrate | 0.3250 | o_height | 45048.5 |
| p | 0.3246 | o_width | 43788.4 |
| size | 0.3244 | bitrate | 7911.08 |
| p_size | 0.3226 | height | 6500.05 |
| frames | 0.3225 | width | 6371.85 |
| i_size | 0.3197 | o_codec | 6265.71 |
| o_width | 0.3117 | p_size | 5589.30 |
| o_height | 0.3116 | size | 5366.45 |
| duration | 0.3015 | i_size | 3458.90 |
| i | 0.3009 | framerate | 2775.40 |
| o_codec | 0.2811 | o_framerate | 987.06 |
| framerate | 0.1676 | o_bitrate | 549.18 |
| height | 0.1488 | frames | 494.39 |
| weidth | 0.1461 | p | 493.60 |
| codec | 0.06637 | i | 383.97 |
| o_bitrate | 0.01799 | duration | 19.1305 |
| o_framerate | 0.01301 | codec | 5.9073 |
| b | 0.00028 | b | 0.2128 |

By sorting features using decreasing order of mutual information and F score, we notice that the feature *bitrate* is an important feature since it has high value for both mutual information and F score, and the feature *b* is an unimportant feature since it has low value for both mutual information and F score.

However, as for other features, they might have a high value for one of these two metrics and a low value for another, making it difficult to judge whether or not they are important features.

To further investigate how the feature selection step might affect performance of our model in terms of test RMSE, we use "SelectKBest" function from "sklearn.feature_selection" library to change the number of selected features, and use the subset of features with linear regression model, to see how the test RMSE would change with regard to the number of selected features with score function being either mutual information or F score.

### Bike Dataset

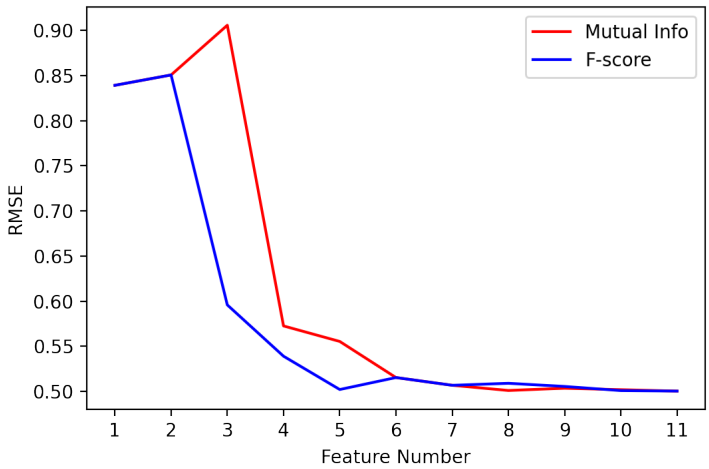



Figure 13: Test RMSE vs feature number for bike dataset

Minimum test RMSE = 0.5003 at feature number = 11 selected by F-score for "SelectKBest" function.

This means all the features for bike dataset is used for regression. Therefore, all the features are important features, though the curve is already almost stable after feature number reaches about 6.

### Suicide Dataset

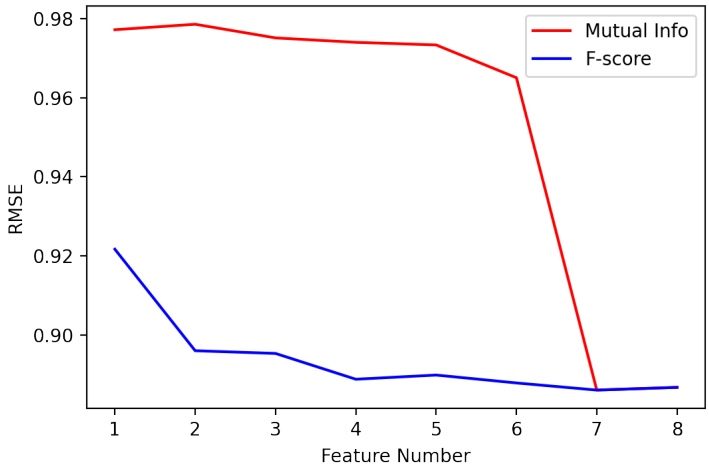



Figure 14: Test RMSE vs feature number for suicide dataset

Minimum test RMSE = 0.8861 at feature number = 7 selected by F-score for "SelectKBest" function.

The 7 most important features selected by the function "SelectKBest" are: {country, sex, age, population, gdp_for_uear, gdp_per_capita, generation}.

**Video Dataset**

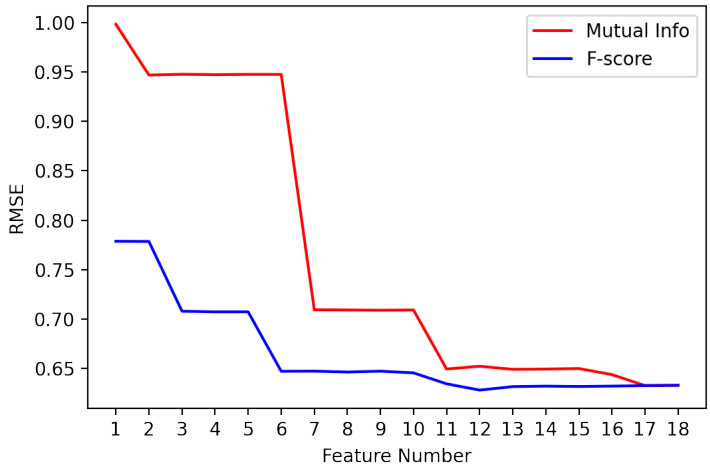



Figure 15: Test RMSE vs feature number for video dataset

Minimum test RMSE = 0.6280 at feature number = 12 selected by F-score for "SelectKBest" function.

The 12 selected most important features are: {width, height, bitrate, framerate, i_size, p_size, size, o_codec, o_bitrate, o_framerate, o_width, o_height}.

## 3.2 Training

### 3.2.1 Linear Regression

**QUESTION 10:** Explain how each regularization scheme affects the learned hypotheses.

Linear Regression fits a linear model with coefficients $w = (w_1, \ldots, w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation[1].

**Linear Regression without regularization:** The objective function is:

$$L(x) = \frac{1}{N}\sum_{i}^{N} \|y_i - w * x_i\|_2^2 \tag{1}$$

When there is no regularization term, minimizing loss function tend to lead to large $w$, which causes overfitting. Regularization ensures that for every extra training parameter, a penalization is given to prevent the model from overfitting to the training dataset

**Lasso (L1 penalty):** The objective function is:

$$L(x) = \frac{1}{N}\sum_{i}^{N} \|y_i - w * x_i\|_2^2 + \alpha\|w\|_1 \tag{2}$$

This regularization encourages sparsity in trained parameters $w$. It does this by adding a regularization penalty in $L_1$ norm to the loss function during training. It penalizes a model based on the sum of the absolute coefficient values[2].

**Ridge (L2 penalty):** The objective function is:

$$L(x) = \frac{1}{N}\sum_{i}^{N} \|y_i - w * x_i\|_2^2 + \alpha\|w\|_2^2 \tag{3}$$

This is a regularization that encourages a small value in trained weight $w$. It has the effect of shrinking the coefficients for those input variables that do not contribute much to the prediction task. This model is based on the sum of the squared coefficient values[3].

**QUESTION 11:** Report the choice of the best regularization scheme along with the optimal penalty parameter and briefly explain

**1. Bike Dataset**

**Ordinary Linear Regression**

Train RMSE: 0.443414
Test RMSE: 0.500304

**Lasso Regression**

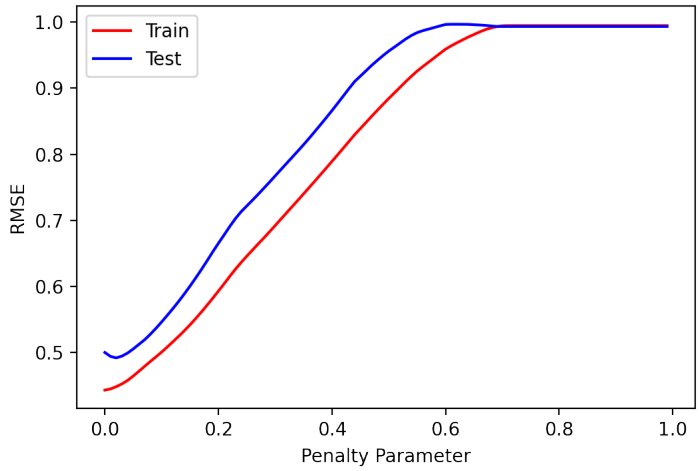



Figure 16: RMSE vs penalty parameter $\alpha$ in Lasso Regression for Bike dataset

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 0.443414, test RMSE = 0.500288
Minimum test RMSE at: $\alpha = 0.02$, train RMSE = 0.448608, test RMSE = 0.492163

**Ridge Regression**

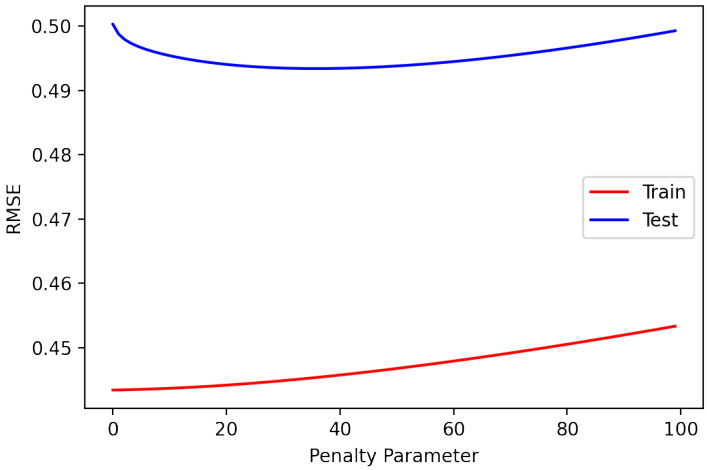



Figure 17: RMSE vs penalty parameter $\alpha$ in Ridge Regression for Bike dataset

Minimum train RMSE at: $\alpha = 0$, train RMSE = 0.443414, test RMSE= 0.500304
Minimum test RMSE at: $\alpha = 36$, train RMSE = 0.445389, test RMSE= 0.493413

**2. Suicide Dataset**

### Ordinary Linear Regression

Train RMSE: 0.872861
Test RMSE: 0.886748

### Lasso Regression

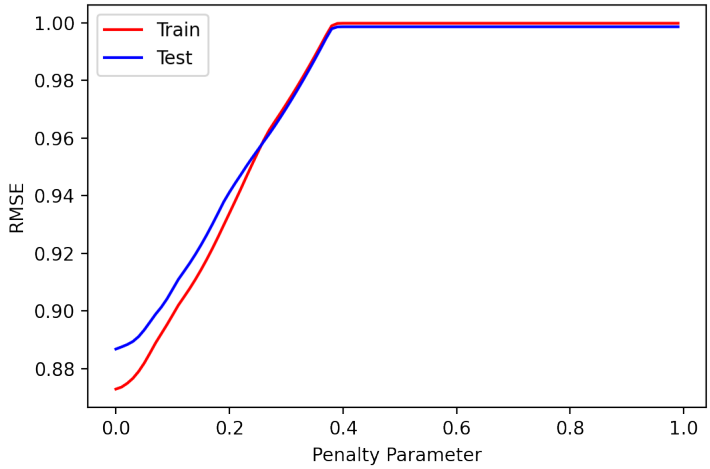



Figure 18: RMSE vs penalty parameter $\alpha$ in Lasso Regression for Suicide dataset

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 0.872861, test RMSE = 0.886748
Minimum test RMSE at: $\alpha = 0.0$, train RMSE = 0.872861, test RMSE = 0.886748

### Ridge Regression

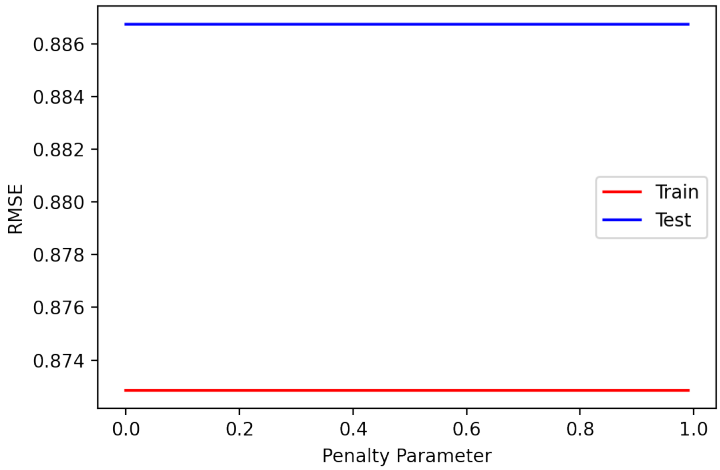



Figure 19: RMSE vs penalty parameter $\alpha$ in Ridge Regression for Suicide dataset

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 0.872861, test RMSE = 0.886748
Minimum test RMSE at: $\alpha = 0.99$, train RMSE = 0.872861, test RMSE = 0.886747

## 3. Video Dataset

### Ordinary Linear Regression

Train RMSE: 0.623483
Test RMSE: 0.632814

### Lasso Regression

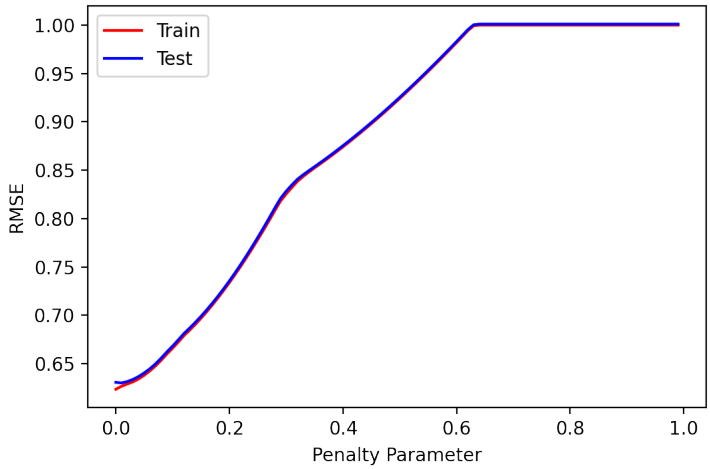



Figure 20: RMSE vs penalty parameter $\alpha$ in Lasso Regression for Video dataset

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 0.623525, test RMSE = 0.630637
Minimum test RMSE at: $\alpha = 0.01$, train RMSE = 0.626607, test RMSE = 0.630054

### Ridge Regression

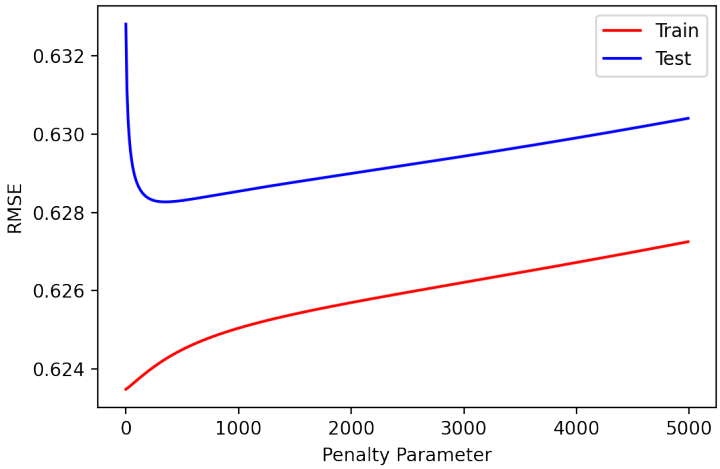



Figure 21: RMSE vs penalty parameter $\alpha$ in Ridge Regression for Video dataset

Minimum train RMSE at: $\alpha = 0$, train RMSE = 0.623483, test RMSE = 0.632814
Minimum test RMSE at: $\alpha = 350$, train RMSE = 0.624252, test RMSE = 0.628271

In conclusion, we have put all the results from three linear regression with optimized penalty parameters in the table together:

| Dataset | LR method | $\alpha$ | train RMSE | test RMSE |
|---|---|---|---|---|
| bike | Ordinary | - | 0.443414 | 0.500304 |
| | Lasso | 0.02 | 0.448608 | **0.492163** |
| | Ridge | 36 | 0.445389 | 0.493413 |
| suicide | Ordinary | - | 0.872861 | 0.886748 |
| | Lasso | 0.0 | 0.872861 | 0.886748 |
| | Ridge | 0.99 | 0.872861 | **0.886747** |
| video | Ordinary | - | 0.623483 | 0.632814 |
| | Lasso | 0.01 | 0.626607 | 0.630054 |
| | Ridge | 350 | 0.624252 | **0.628271** |

Table 7: Optimized penalty parameters

We notice that, Ridge Regression seems to be a marginally better regularization scheme as RMSE is decreasing for the most part and later becoming constant. Also, empirically, L2 regularization generally works better compared to L1.

Another observation is that RMSE decreases as the penalty parameters for Ridge Regression increases to a level, but then increases after that, suggesting that there exists some trade-off for selecting proper value of $\alpha$. By sweeping this hyper-parameter over a proper range, we can find an optimized $\alpha$ using validation set from a given dataset.

**QUESTION 12:**

Feature Scaling is a method used to normalize the range of independent variables or features of data[6].

Standardization involves rescaling the features such that they have the properties of a standard normal distribution with a mean of zero and a standard deviation of one[7]. This helps balancing the weight between different features, which is also proved by our experiments on three given datasets.

Our experiments in Question 11 are based on features after standardization, in order to prove its effects, we are going to train and test on features without standardization in Question 12.

**1. Bike Dataset**

**Ordinary Linear Regression**

Train RMSE: 858.399295
Test RMSE: 968.531165

### Lasso Regression

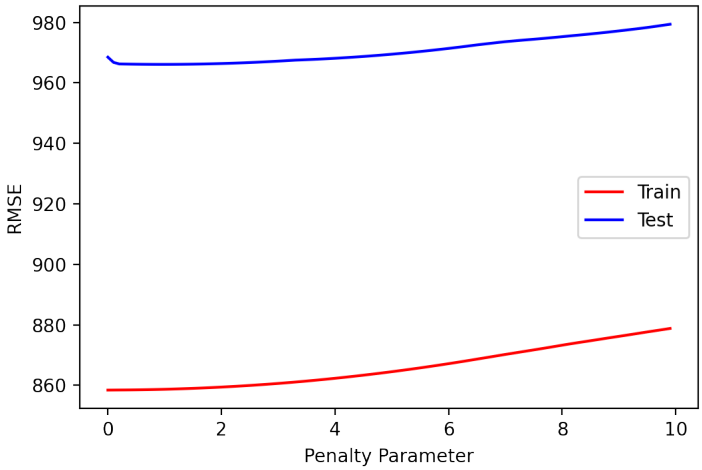



Figure 22: RMSE vs $\alpha$ in Lasso Regression of Bike dataset (without standardization)

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 858.399298, test RMSE = 968.499572
Minimum test RMSE at: $\alpha = 1.0$, train RMSE = 858.664325, test RMSE = 966.111535

### Ridge Regression

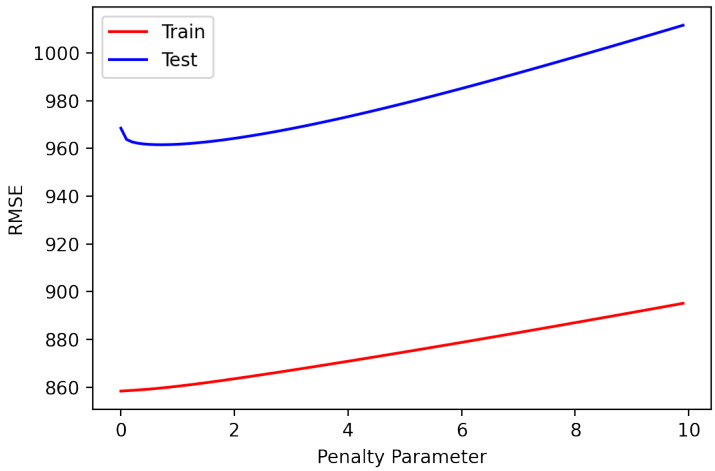



Figure 23: RMSE vs $\alpha$ in Ridge Regression of Bike dataset (without standardization)

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 858.399295, test RMSE = 968.531165
Minimum test RMSE at: $\alpha = 0.7$, train RMSE = 859.661312, test RMSE = 961.599527

### 2. Suicide Dataset

### Ordinary Linear Regression

Train RMSE: 1.133376
Test RMSE: 1.151408

### Lasso Regression

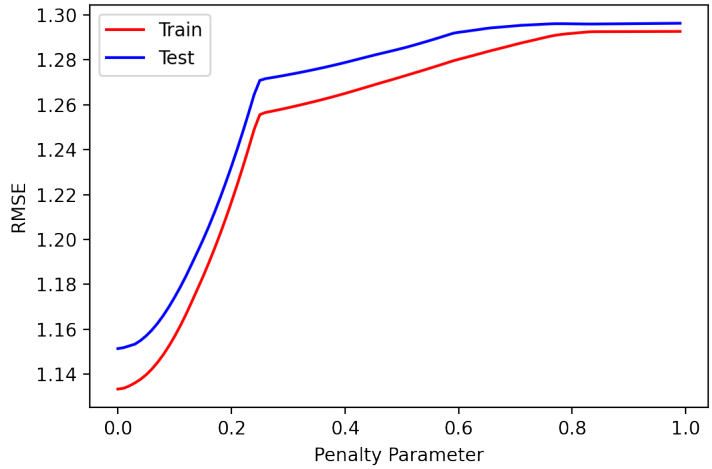



Figure 24: RMSE vs $\alpha$ in Lasso Regression of Suicide dataset (without standardization)

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 1.133376, test RMSE = 1.151408
Minimum test RMSE at: $\alpha = 0.0$, train RMSE = 1.133376, test RMSE = 1.151408

### Ridge Regression

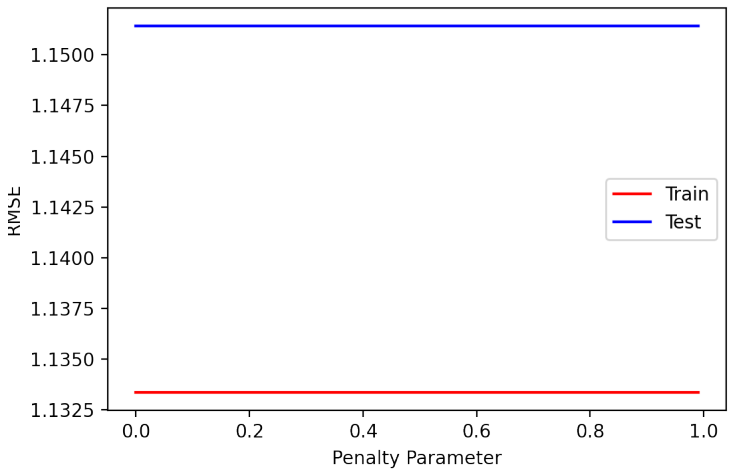



Figure 25: RMSE vs $\alpha$ in Ridge Regression of Suicide dataset (without standardization)

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 1.133376, test RMSE = 1.151408
Minimum test RMSE at: $\alpha = 0.99$, train RMSE = 1.133376, test RMSE= 1.151408

## 3. Video Dataset

### Ordinary Linear Regression

Train RMSE: 0.741666
Test RMSE: 0.752765

### Lasso Regression

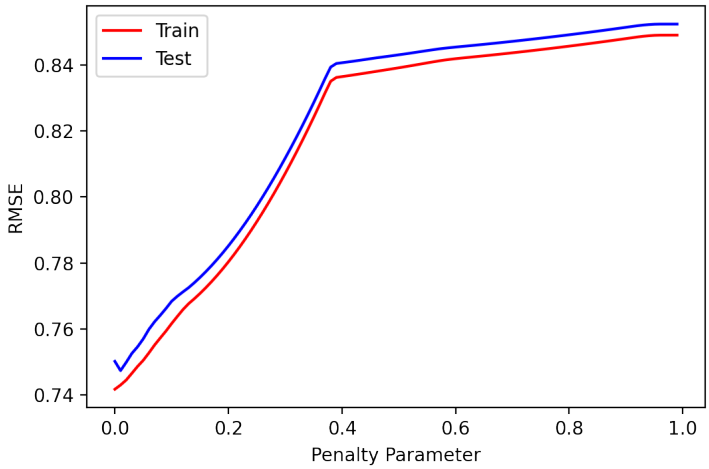



Figure 26: RMSE vs $\alpha$ in Lasso Regression of Video dataset (without standardization)

Minimum train RMSE at: $\alpha = 0.0$, train RMSE = 0.741715, test RMSE = 0.750176
Minimum test RMSE at: $\alpha = 0.01$, train RMSE = 0.742992, test RMSE = 0.747357

### Ridge Regression

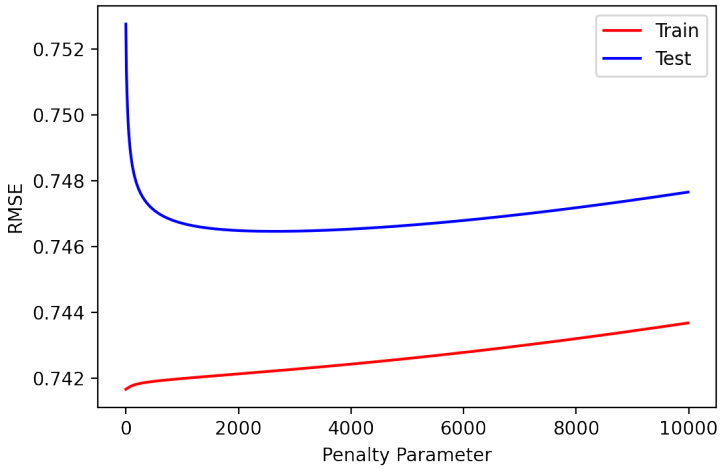



Figure 27: RMSE vs $\alpha$ in Ridge Regression of Video dataset (without standardization)

Minimum train RMSE at: $\alpha = 0$, train RMSE = 0.741666, test RMSE = 0.752765
Minimum test RMSE at: $\alpha = 2640$, train RMSE = 0.742222, test RMSE = 0.746465

In conclusion, we have put all the results from three linear regression with optimized penalty parameters in the table together for comparison:

| Dataset | LR method | alpha | train RMSE | test RMSE |
|---|---|---|---|---|
| bike | Ordinary | - | 858.3993 | 968.5312 |
| | Lasso | 1.0 | 858.6643 | 966.1115 |
| | Ridge | 0.7 | 859.6613 | **961.5995** |
| suicide | Ordinary | - | 1.133376 | 1.1514084 |
| | Lasso | 0.0 | 1.133376 | 1.1514084 |
| | Ridge | 0.99 | 0.133376 | **1.1514081** |
| video | Ordinary | - | 0.741666 | 0.752765 |
| | Lasso | 0.01 | 0.742992 | 0.747357 |
| | Ridge | 2640 | 0.742222 | 0.746465 |

Table 8: Optimized penalty parameters (without standardization)

As we can see from comparison of Table 7 anad Table 8, regression without standardization would cause an enormous increase in the RMSE values, both with and without regularization. Therefore, it is empirically proved that feature scaling in general helps reduce errors.

We also notice that, even though the absolute value of training and test error of Ridge Regression becomes larger compared to before, Ridge Regression is still the best regression model, suggesting that L2 regularization is very powerful and stable to deal with noisy training data that does not follow normal distribution.

**QUESTION 13:**

$p$-Value stands for the probability of weight being zero for the corresponding feature. The feature with smaller $p$-value means it is more significant. So for the significant features, we can sort $p$-value and look for the least $p$-values.

**Bike Dataset**

Table 9 shows the sorted $p$-value from minimum to maximum, which means the feature list from top to bottom is also ordered from the most significant to least significant: {yr, season, weathersit, windspeed, weekday, hum, holiday, mnth, atemp, workingdat, temp}

| feature name | $p$-value |
|---|---|
| yr | 1.373100e-136 |
| season | 1.438629e-19 |
| weathersit | 2.140988e-14 |
| windspeed | 2.899484e-08 |
| weekday | 2.524699e-05 |
| hum | 1.219960e-03 |
| holiday | 9.981321e-03 |
| mnth | 2.266846e-02 |
| atemp | 2.476482e-02 |
| workingday | 9.483725e-02 |
| temp | 1.484887e-01 |

Table 9: $p$-value of features in bike dataset

The best feature here is the feature year as it has the least p-value as shown in the table.

**Suicide Dataset**

The list of features for suicide dataset from most significant to least significant according to $p$-value is as follows: {sex, generation, HDI for year, age, country, gdp_for_year, gdp_per_capita, year, population}.

| feature name | $p$-value |
|---|---|
| sex | 0.000000e+00 |
| generation | 8.291636e-106 |
| HDI for year | 6.702345e-37 |
| age | 1.963506e-33 |
| country | 7.259177e-32 |
| gdp_for_year | 2.842843e-30 |
| gdp_per_capita | 2.761629e-08 |
| year | 1.301551e-04 |
| population | 1.937532e-02 |

Table 10: $p$-value of features in suicide dataset

The best feature here is the feature generation as it has the least p-value as shown in the table.

**Video dataset**

The list of features for video dataset from most significant to least significant according to $p$-value is as follows: {o_framerate, o_codec, o_bitrate, o_height, framerate, p_size, duration, o_width, i_size, i, b, height, size, bitrate, width, frames, o, codec}

The first few results fit intuition since the output framerate, codec, bitrate and image size (in terms of output height and width) would influence the transcoding time most significantly.

The $p$-value for all the features are presented in Table 11.

| feature name | $p$-value |
|---|---|
| o_framerate | 0.000000e+00 |
| o_codec | 0.000000e+00 |
| o_bitrate | 1.030550e-304 |
| o_height | 4.293403e-271 |
| framerate | 3.624523e-33 |
| p_size | 2.316649e-24 |
| duration | 3.219638e-15 |
| o_width | 1.671232e-14 |
| i_size | 1.486288e-13 |
| i | 1.608083e-08 |
| b | 8.142708e-07 |
| height | 5.543802e-05 |
| size | 6.905903e-05 |
| bitrate | 5.059270e-03 |
| width | 9.373526e-03 |
| frames | 1.075348e-01 |
| p | 6.333442e-01 |
| codec | 8.244641e-01 |

Table 11: $p$-value of features in video dataset

The best feature here is the feature o_height as it has the least p-value as shown in the table.

### 3.2.2 Polynomial Regression

**QUESTION 14:**

In order to find the salient polynomial features, we sweep the degree of polynomial *poly_deg* from 1 to 5, and then use "SelectKBest" with either mutual information or F score to find $k$ effective features among the large pool of new artificial features. The reduced number of feature $feature\_num$ is swept from 1 to 30. The train and test RMSE is get by using Linear Regression model to fit and predict the extracted features with 10-fold cross-validation.

### Bike Dataset

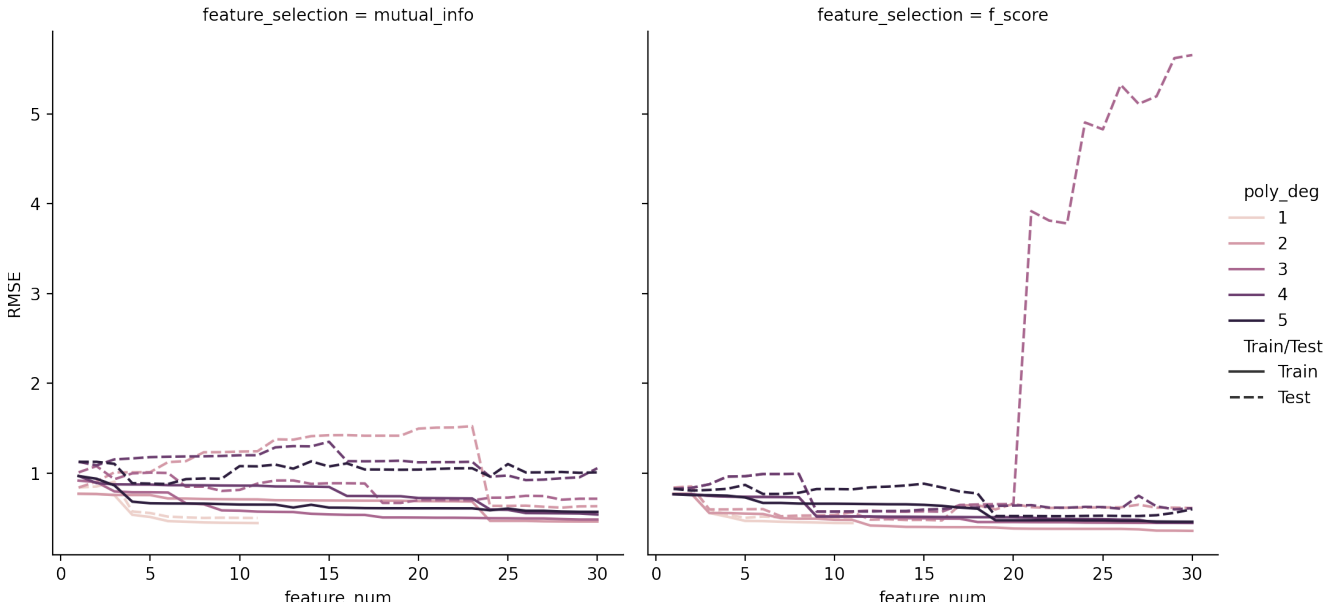



Figure 28: RMSE vs *poly_deg*, *feature_num* for bike dataset

Minimum train RMSE at: poly_deg = 2, feature_num = 30, feature selection method = F score, train RMSE = 0.356801, test RMSE = 0.611537
Minimum test RMSE at: poly_deg = 2, feature_num = 16, feature selection method = F score, train RMSE = 0.398235, test RMSE = 0.472813.

Using poly_deg = 2 and feature_num = 16, we can find the 16 best 2-degree features selected by the training model: {season, yr, mnth, weathersit, temp, atemp, season$^2$, season * temp, season * atemp, yr$^2$, mnth$^2$, mnth * temp, mnth * atemp, weathersit$^2$, temp * atemp, atemp$^2$}.

In Question 9 from Section 3.1.4 and Question 13 from Section 3.2.1, we already experimented with the selection of 1-degree significant features, and we have already noticed that the feature *atemp* and *temp* are closely related to the target, and here when we are trying to select 2-degree features, it is no coincidence that the appearance of *atemp* and *temp* are quite frequent in the factor. We can also see features like $atemp * atemp$, $temp * atemp$, which indicate that polynomial regression is in some way an extended version linear regression, and many of the selected salient features would collide with that extracted from linear regression.

But we also observe the phenomenon of overfitting, when feature number and polynomial degree is too large. From the F_score figure in Fig. 28, the training error is continuously dropping when feature number increases, but the test error can suddenly increase, indicating that the training model is overfitting at that point.

### Suicide Dataset

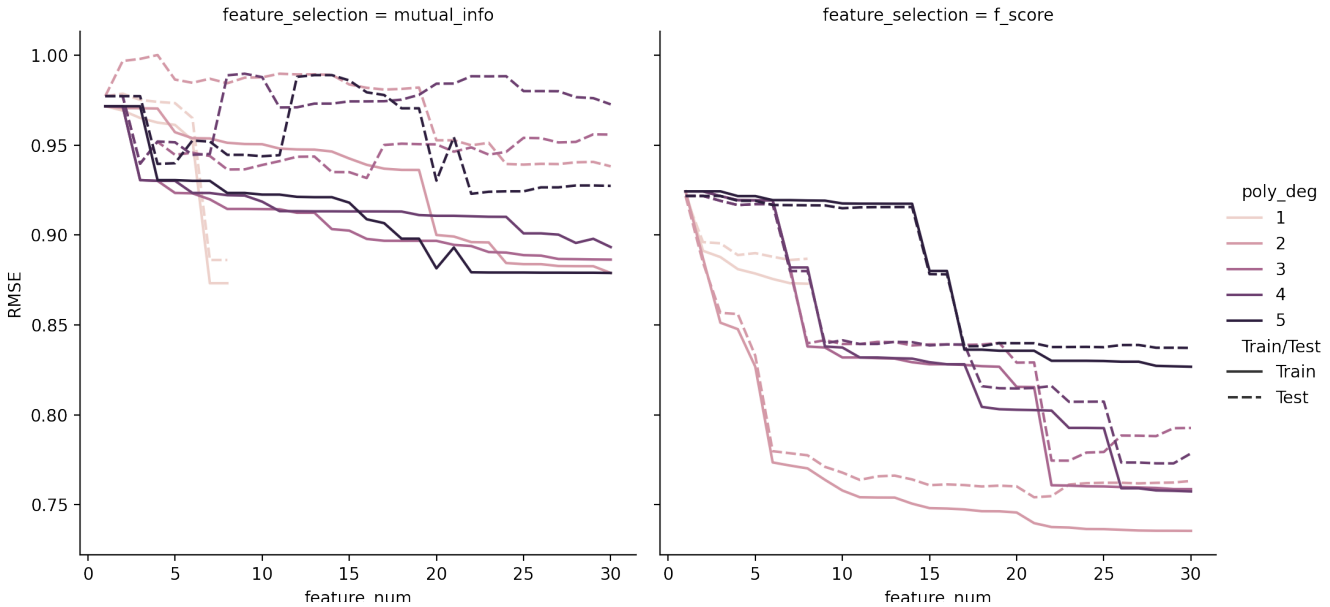



Figure 29: RMSE vs *poly_deg*, *feature_num* for suicide dataset

Minimum train RMSE at: poly_deg = 2, feature_num = 30, feature selection method = F score, train RMSE = 0.735422, test RMSE = 0.763188
Minimum test RMSE at: poly_deg = 2, feature_num = 21, feature selection method = F score, train RMSE = 0.739728, test RMSE = 0.754113

The 21 features selected by "SelectKBest" function based on the 2-degree polynomial feature are as follows: {country, sex, age, population, gdp_for_year, gdp_per_capita, generation, country * population, country * gdp_for_year, country * gdp_per_capita, year * gdp_for_year, year * gdp_per_capita, year * generation, age$^2$, age * gdp_for_year, age * generation, population$^2$, population * gdp_for_year, population * gdp_per_capita, gdp_for_year$^2$, generation$^2$}.

Many of the selected polynomial features actually have practical meaning, for example, the feature (population * gdp_per_capita) is selected, meaning the total gdp of the year. In many features, *gdp_for_year* and *gdp_per_capita* appear symmetrically, which also makes sense since the two values represent similar economic feature, and therefore if one of them is a salient feature, the other has a high probability of also being selected.

### Video Dataset

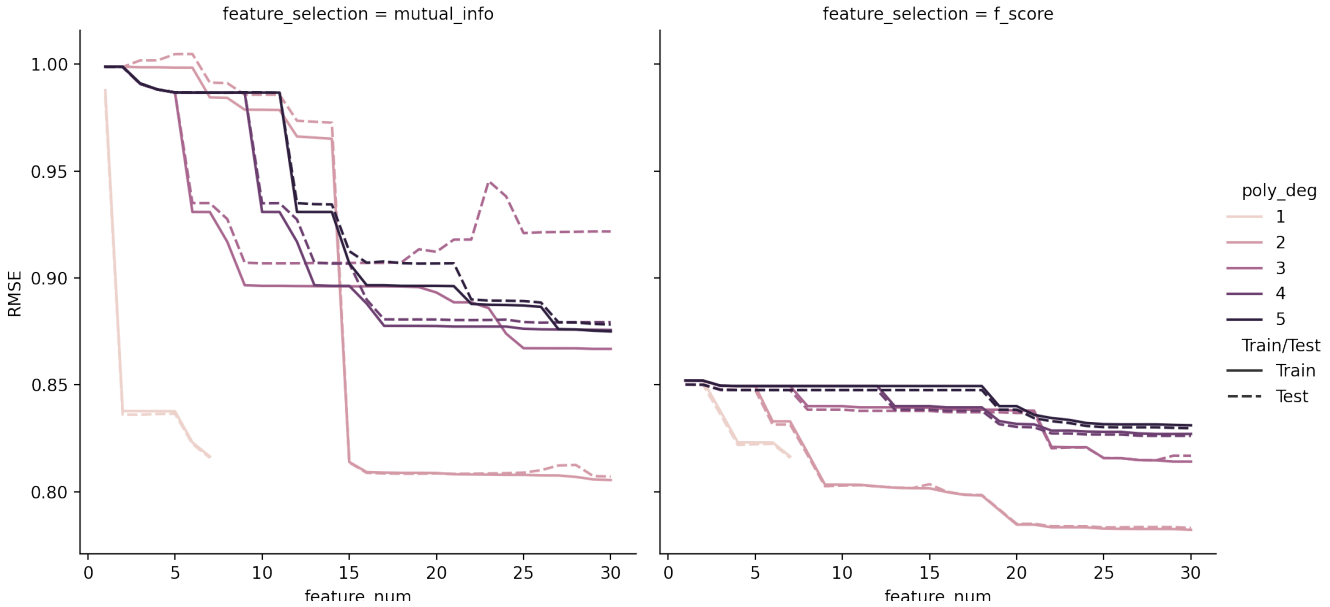



Figure 30: RMSE vs *poly_deg*, *feature_num* for video dataset

Minimum train RMSE at: poly_deg = 2, feature_num = 30, feature selection method = F score, train RMSE = 0.782136, test RMSE = 0.783054
Minimum test RMSE at: poly_deg = 2, feature_num = 30, feature selection method = F score, train RMSE = 0.782136, test RMSE = 0.783054

The 30 features selected by "SelectKBest" function based on the 2-degree polynomial features are as follows: {width, height, bitrate, framerate, i_size, p_size, size, o_codec, o_framerate, o_width, o_height, width$^2$, width * height, width * bitrate, width * p_size, width * size, width * o_width, width * o_height, height$^2$, height * bitrate, height * p_size, height * size, height * o_width, height * o_height, framerate$^2$, framerate * i_size, o_codec$^2$, o_width, o_width * o_height, o_height$^2$}.

We notice many of the selected features for video dataset are size-related: there is a great portion of selected features in the top 30 features include at least one of these 4 features {width, height, o_width, o_height}. This makes sense and fit our common sense intuision since the input and output of the image scale would influence the transcoding time greatly.

### QUESTION 15:

In order to find a proper degree of polynomial and also search the parameter grid efficiently, we fix the feature_num = 2, feature selection method = F score, and change poly_deg and regularization factor for Lasso and Ridge Regression.

The reason why we search poly_deg and different regularization factor together is that, regulariza-

tion is closely related to overfitting, which would be easily induced by large poly_deg. Therefore, changing them together might hopefully balance the overfitting while still increasing the accuracy.

The experiment results in this question for each dataset would be shown in two figures: the first one is RMSE vs regularization factor $\alpha$ for Lasso Regression and Ridge Regression respectively, the second one is RMSE vs poly_deg where each point on the curve chooses the $\alpha$ that has the best test performance.

### Bike Dataset

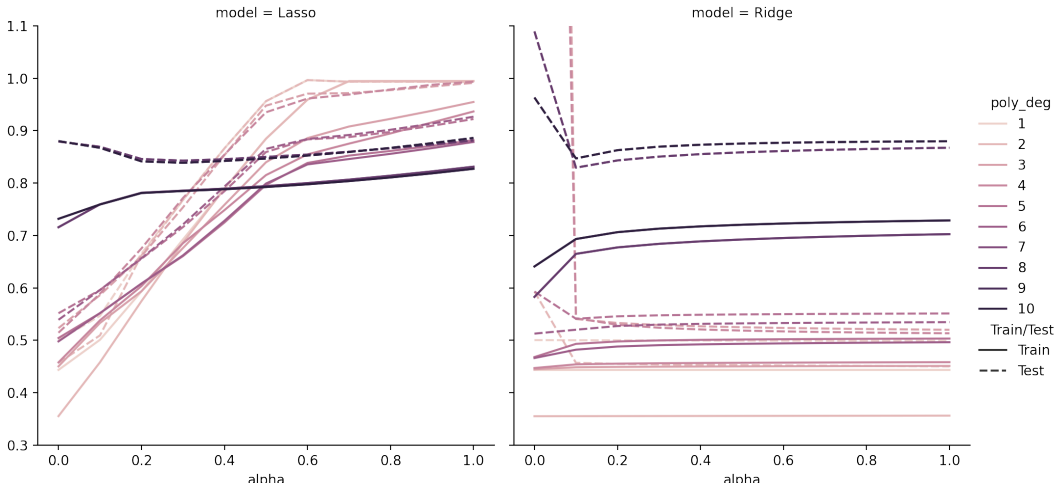



Figure 31: RMSE vs regularization factor $\alpha$ for bike dataset

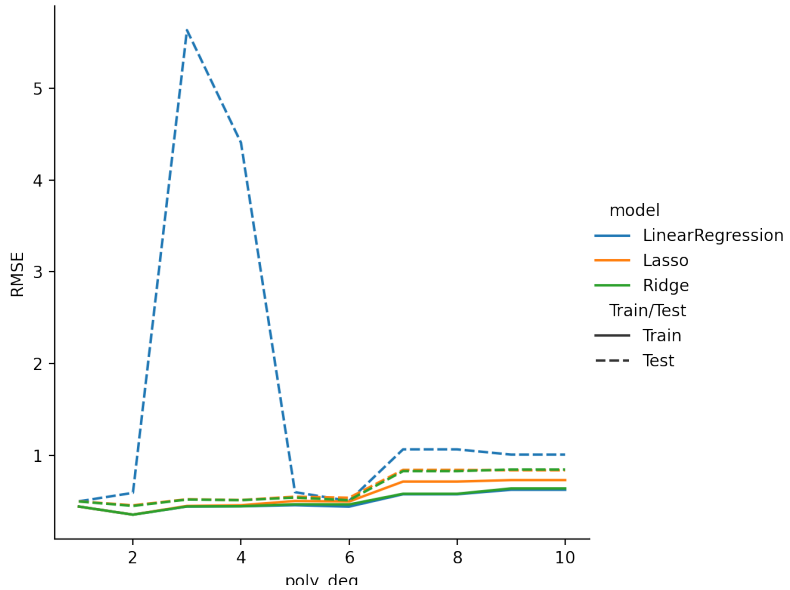



Figure 32: RMSE vs degree of polynomial for bike dataset

Minimum train RMSE at: poly_deg = 2, regression model = Ridge, $\alpha = 0$, train RMSE = 0.355066, test RMSE = 0.592906
Minimum test RMSE at: poly_deg = 2, regression mode = Ridge, $\alpha = 1$, train RMSE = 0.356189, test RMSE = 0.450201.

From Fig. 32, we can see that with the increase of polynomial degree, the test error can be very unstable using Linear Regression, while Ridge Regression model almost always produces the best performance. The overall best performance comes at polynomial degree = 2 with Ridge Regression model using $\alpha = 1$.

### Suicide Dataset

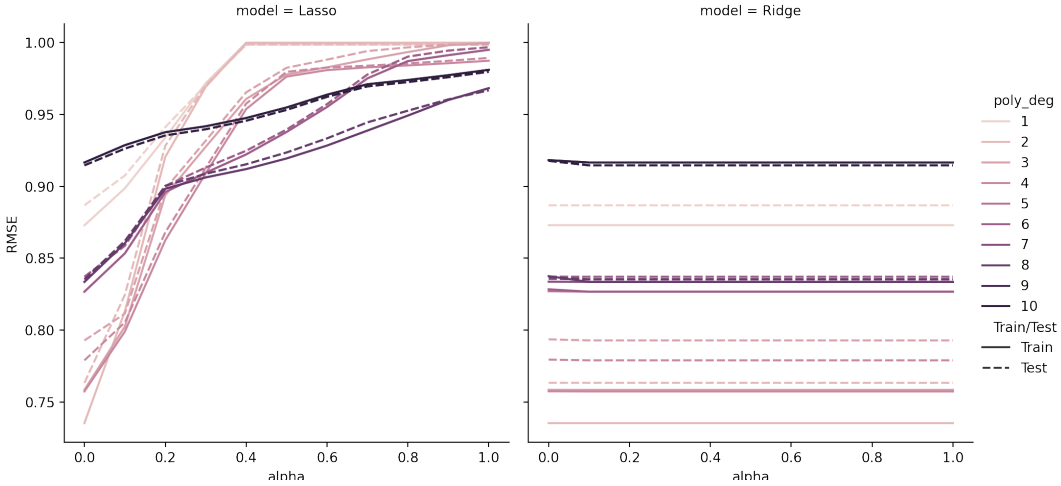



Figure 33: RMSE vs regularization factor $\alpha$ for suicide dataset

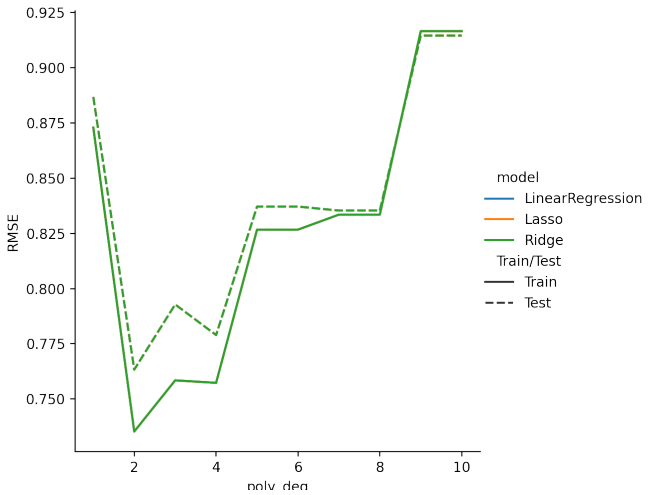



Figure 34: RMSE vs degree of polynomial for suicide dataset

Minimum train RMSE at: poly_deg = 2, regression model = Linear Regression, train RMSE = 0.735287, test RMSE = 0.763343
Minimum test RMSE at: poly_deg = 2, regression mode = Lasso, $\alpha = 0$, train RMSE = 0.735302, test RMSE = 0.763262.

Fig. 34 shows very clear that the minimum RMSE only occurs when polynomial degree = 2: if the degree is less, selected features are not sufficient for regression, leading to underfitting; if the degree is more, it would become overfitting.

Fig. 33 also shows that the best performance for Lasso Regression with whatever poly_deg is at $\alpha$=0, which returns to Linear Regression. And Ridge Regression is almost never influenced by the change of $\alpha$, which again could return to Linear Regression model. This might be caused due to the fact that the feature space has a polynomial degree of two while the Ridge Regression also applies 2-degree regularization term, there is a possibility that with proper set-up the two might be cancelled. But this assumption would still need further investigate.

### Video Dataset

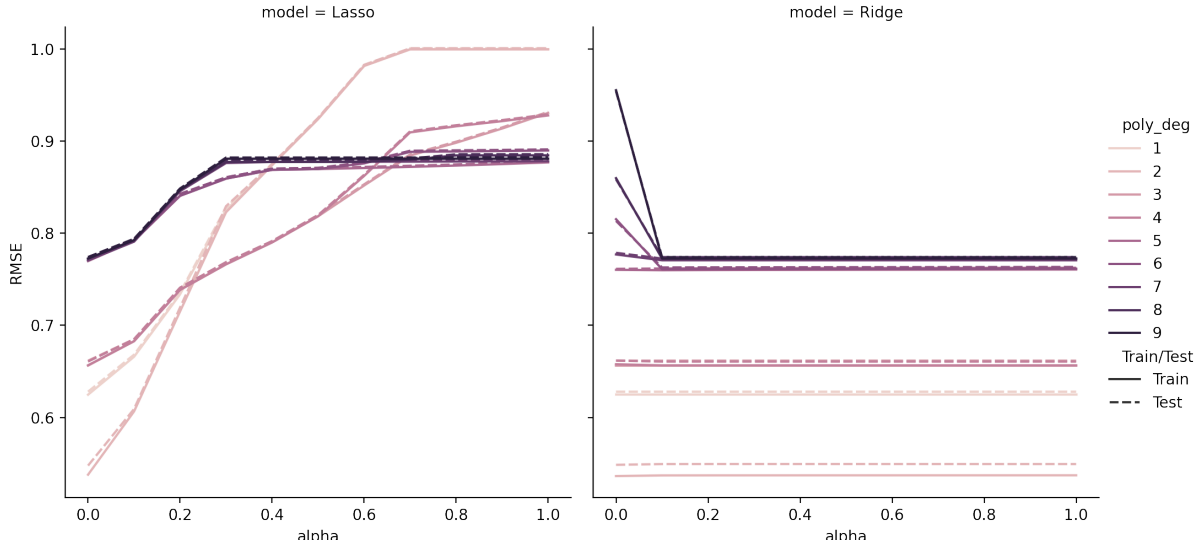



Figure 35: RMSE vs regularization factor $\alpha$ for video dataset

Minimum train RMSE at: poly_deg = 2, regression model = Ridge, $\alpha = 0$, train RMSE = 0.536315, test RMSE = 0.548532
Minimum test RMSE at: poly_deg = 2, regression mode = Lasso, $\alpha = 0$, train RMSE = 0.537817, test RMSE = 0.547688

Similar to suicide dataset, Fig. 36 also has a sharp drop at poly_deg = 2, showing that this is the best value for degree of polynomial.

The optimized value of $\alpha$ for Lasso Regression and Ridge Regression is zero again, indicating that

after the 2-degree features are selected, simple linear regression would be sufficient to solve the regression problem without having to worry about overfitting.

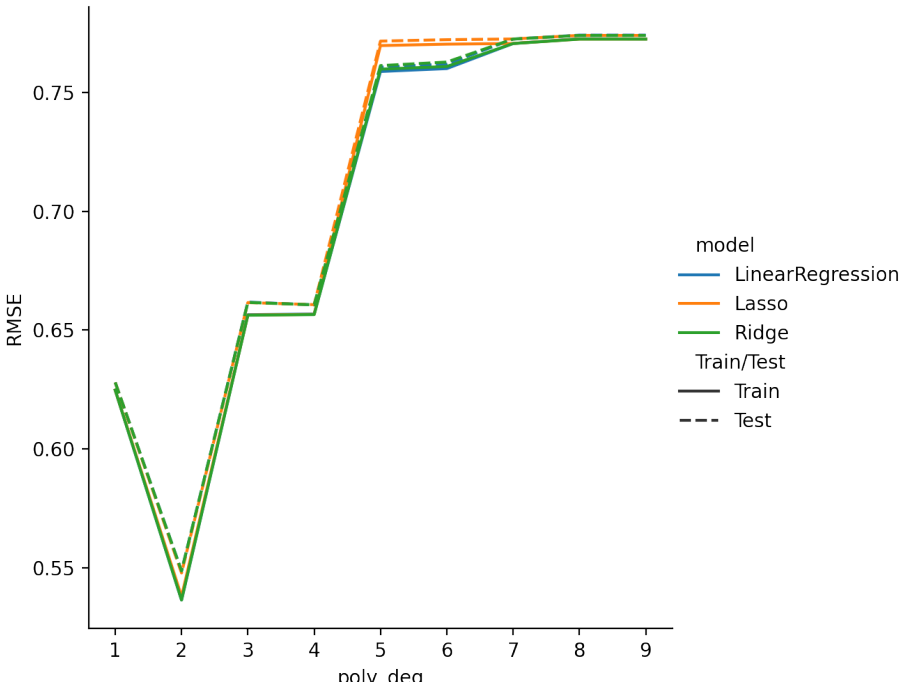



Figure 36: RMSE vs degree of polynomial for video dataset

To conclude, for all three datasets, the best degree of polynomial is all 2. The reason why we shouldn't increase the polynomial degree too much is that we want to avoid overfitting.

**QUESTION 16:**

For video transcoding dataset, it might make sense to craft inverse of certain features such that we get features such as $\frac{x_i x_j}{x_k}$, etc. The reason why it might make sense is that many of the raw features have practical meaning after such operation, such as $width * height$ means the area of the input frame, $o\_width/width$ means the changing ratio of the image width, similar relationship holds for video duration, framerate, and total frame number, etc.

Since $\frac{x_i x_j}{x_k}$ can be view as a 3-degree polynomial from feature list $\{x_i, x_j, \frac{1}{x_k}\}$. Therefore, for implementation, we extend the raw feature space $\{x_0, x_1, ..., x_n\}$ to $\{x_0, x_1, ..., x_n, 1/x_0, 1/x_1, ..., 1/x_n\}$, use poly_deg = 3 and follow the same process as in Question 14 to search for the best feature_num using Linear Regression model. The experiment result is plotted in Fig. 37.

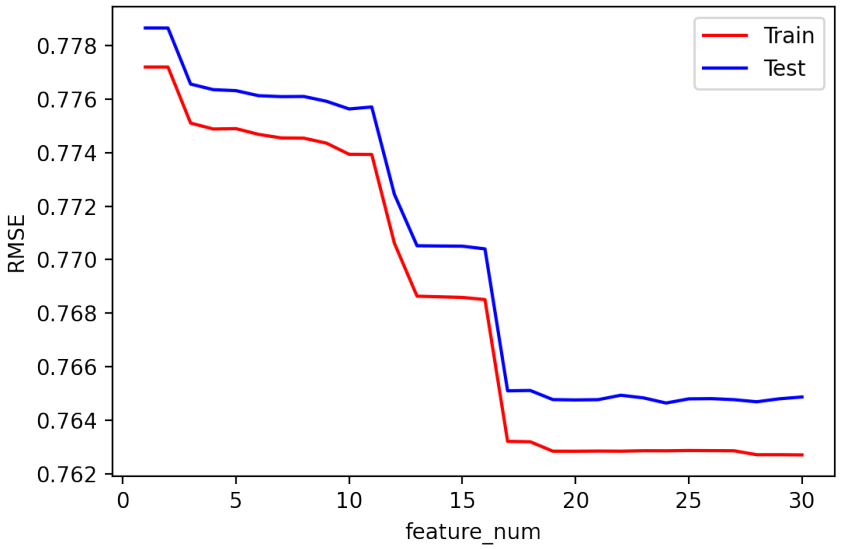



Figure 37: RMSE vs $feature\_num$ for video dataset with inversed feature

Minimum train RMSE at: feature_num = 30, train RMSE = 0.762716, test RMSE = 0.764877
Minimum test RMSE at: feature_num = 24, train RMSE = 0.762869, test RMSE = 0.764652

The top 16 features selected by "SelectKBest" function are: {o_width, o_height, $\frac{\text{o\_width}}{\text{o\_height}}$, $\frac{\text{o\_height}}{\text{o\_width}}$, $\frac{\text{frames*o\_width}}{\text{p}}$, $\frac{\text{frames*o\_height}}{\text{p}}$, $\text{o\_width}^3$, $\text{o\_width}^2 * \text{o\_height}$, $\text{o\_width}^2 * \text{o\_height}$, $\frac{\text{o\_width}}{\text{o\_framerate}^2}$, $\text{o\_height}^3$, $\frac{\text{o\_height}^2}{\text{o\_width}}$, $\frac{\text{o\_height}}{\text{o\_framerate}^2}$, $\frac{\text{o\_height}}{\text{o\_width}^2}$, $\frac{1}{\text{o\_width}}$, $\frac{1}{\text{o\_height}}$ }

We notice a large portion of the selected features are still based on *o_width* and *o_height*, which once again emphasize their importance. There are also terms that look similar to the example of $\frac{x_i x_j}{x_k}$, however, the performance is not boosted.

The baseline results with Linear Regression and poly_deg = 2 is: train RMSE = 0.5365, test RMSE = 0.5487, both of which is smaller than the current best results. A potential reason might be that, with larger feature space to search from, the model might not be able to select the most sufficient features. For example, in our case, the "SelectKBest" function also returns quite a lot redundant features such as $\frac{\text{o\_width}}{\text{o\_width}}$, $\frac{\text{codec} * \text{o\_width}}{\text{codec}}$, etc. Therefore, larger $feature\_num$ might be needed in order to get a better performance, as well as using Ridge Regression to replace the Linear Regression model.

### 3.2.3 Neural Network

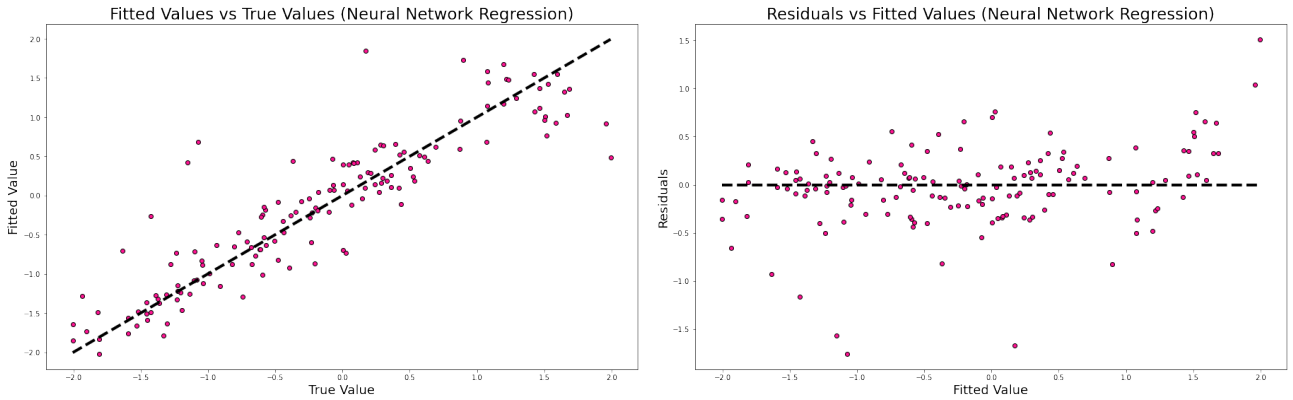



Figure 38: Neural Network - Bike Dataset

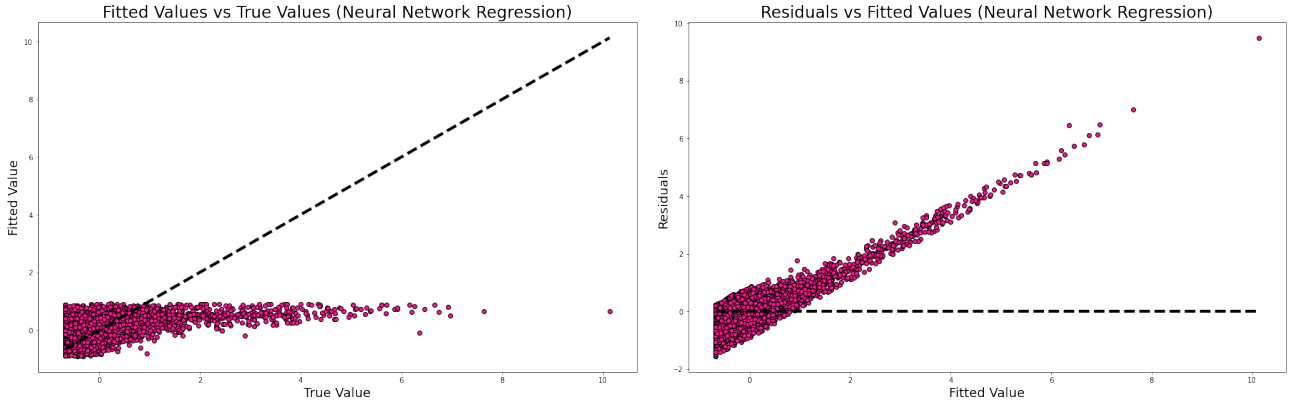



Figure 39: Neural Network - Suicide Dataset

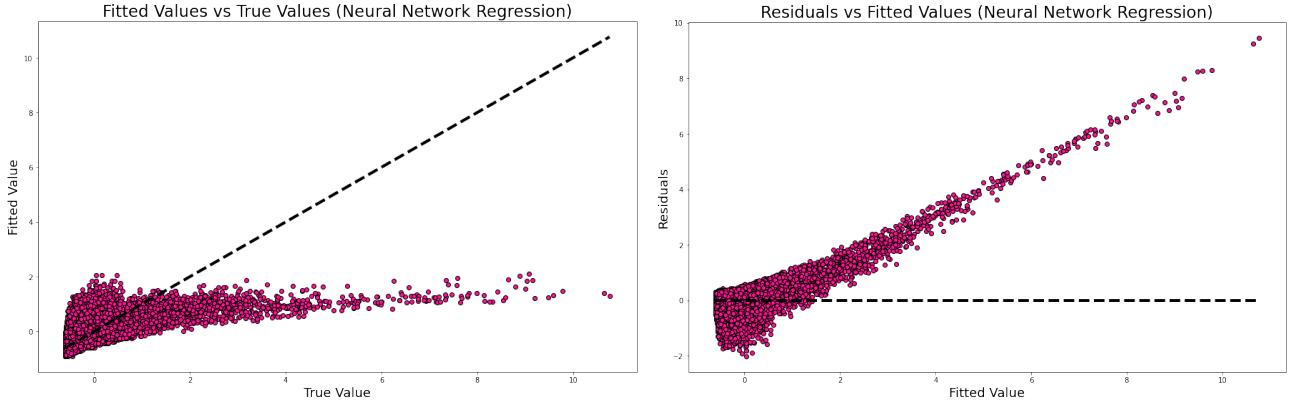



Figure 40: Neural Network - Video Dataset

**QUESTION 17: Why does it do much better than linear regression**

Neural Network does better for the dataset where the relation between the outcome and the predictors is non-linear. Linear regression will do better when the underlying relationship between the variables and the response is known to be linear. But from the preprocessing stage we can see, the correlation between the dependent variables and the independent variable are low in absolute values for some datasets. This means the underlying relationship between the variables are not linear. Hence, neural network does better.

**QUESTION 18:** The parameter space we searched is listed below.

- hidden layer sizes: [(64, 64, 64), (50,100,50), (128,1)]

- alpha: [0.0001, 0.001, 0.01, 0.05]

**The best parameter combination for Bike dataset**

- alpha: 0.001

- hidden layer sizes: (50, 100, 50)

- Average Train RMSE: 0.4679

- Average Test RMSE: 0.5020

**The best parameter combination for Suicide dataset**

- alpha: 0.05

- hidden layer sizes: (50, 100, 50)

- Average Train RMSE: 0.8940

- Average Test RMSE: 0.8868

**The best parameter combination for Video dataset**

- alpha: 0.01

- hidden layer sizes: (64, 64, 64)

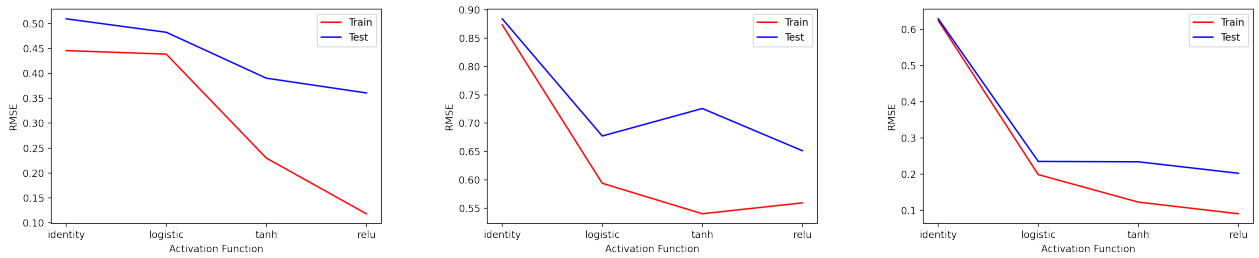
- Average Train RMSE: 0.8164

- Average Test RMSE: 0.8159

**QUESTION 19:**

We should not use activation function for outer layer because this is linear regression problem and we are interested in numerical values without any transformation.

For the hidden layer, we experimented with three different kinds of activation functions: logistic, tanh, relu, and compared with identity (which means none of the activation function is used).




(a) Bike dataset (b) Suicide dataset (c) Video dataset

Figure 41: Affects of activation functions

For each dataset, all the rest parameters are using the optimized parameters that we previously found. As we can see from Fig. 41, for all the datasets, "relu" always returns the best performance, so we are going to use relu as our activation function for the hidden layer.

**QUESTION 20:**

Too much increase of the depth of the network will cause overfitting. If we build a very deep network, we might have each layer just memorize the output so the neural network will fail to generalize to new data.

### 3.2.4 Random Forest

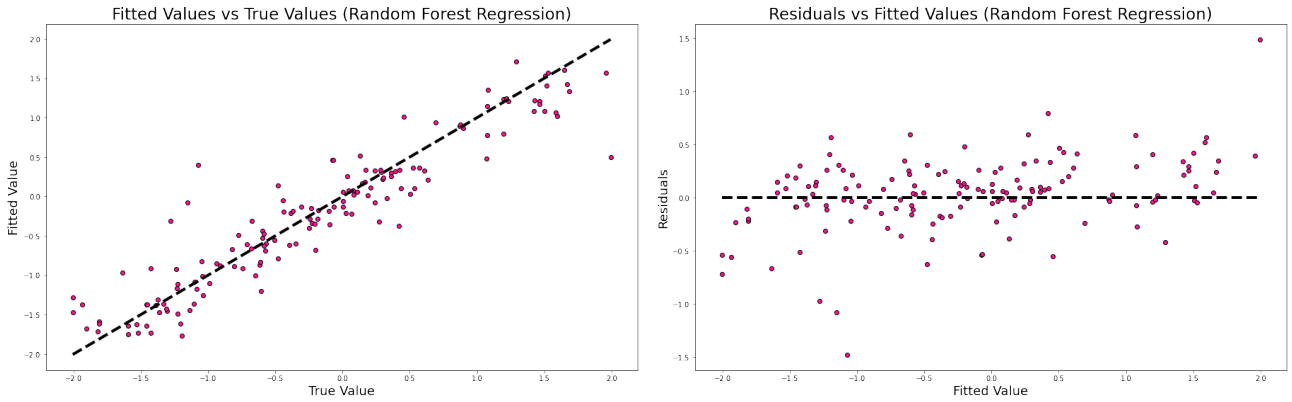



Figure 42: Random Forest - Bike Dataset

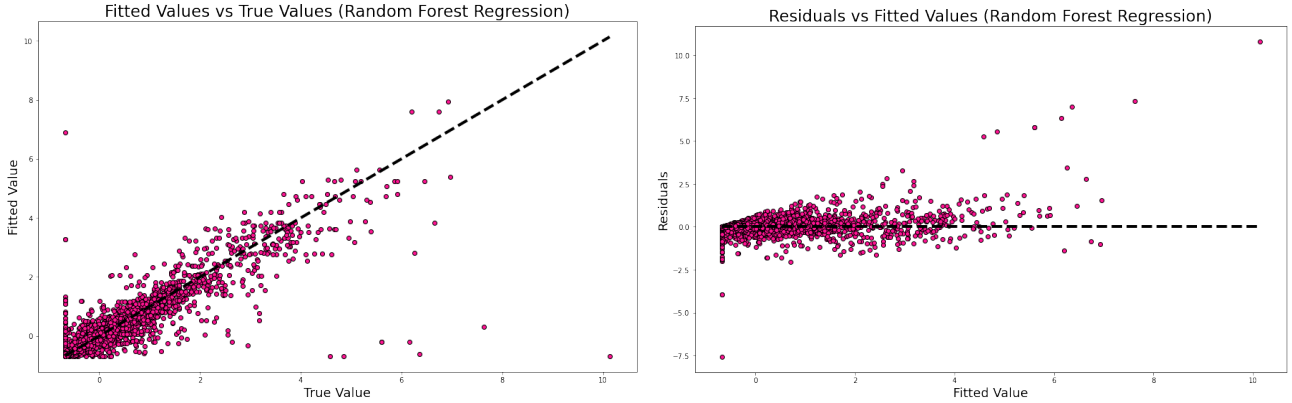



Figure 43: Random Forest - Suicide Dataset

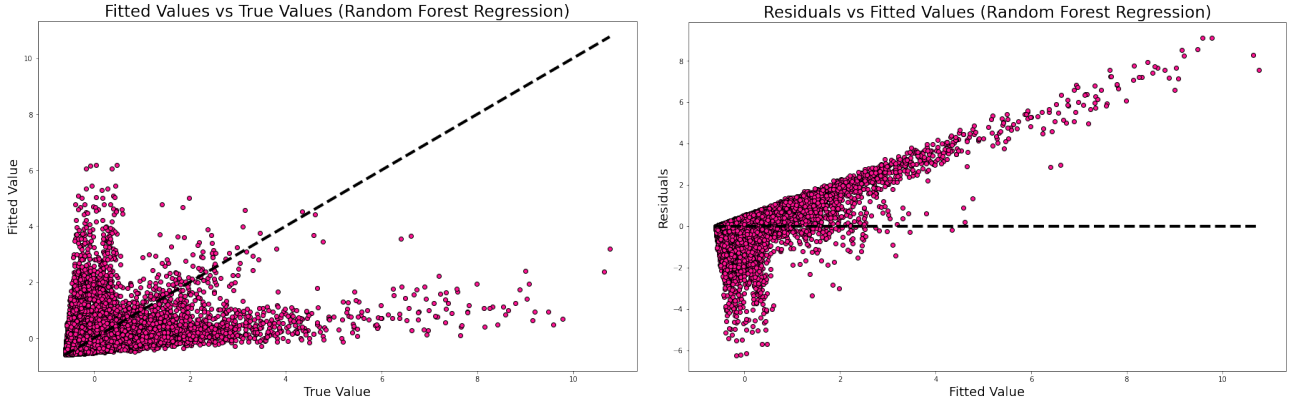



Figure 44: Random Forest - Video Dataset

The parameter space we searched is listed below.

- max depth: [80, 90, 100, 110],

- max features: [2, 3, 5],

- n estimators': [100, 200, 300, 1000]

**The best parameter combination for Bike dataset**

- max depth: 110,

- max features: 2,

- n estimators: 300

- Average Train RMSE: 0.1534

- Average Test RMSE: 0.4957

- Out of bag error: 0.8218

**The best parameter combination for Suicide dataset**

- max depth: 90,

- max features: 3,

- n estimators: 100

- Average Train RMSE: 0.3145

- Average Test RMSE: 1.0111

- Out of bag error: 0.8783

**The best parameter combination for Video dataset**

- max depth: 110,

- max features: 2,

- n estimators: 1000

- Average Train RMSE: 0.7524

- Average Test RMSE: 0.8071

- Out of bag error: -0.014

**QUESTION 21:** The following experiments are done in the Bike dataset.

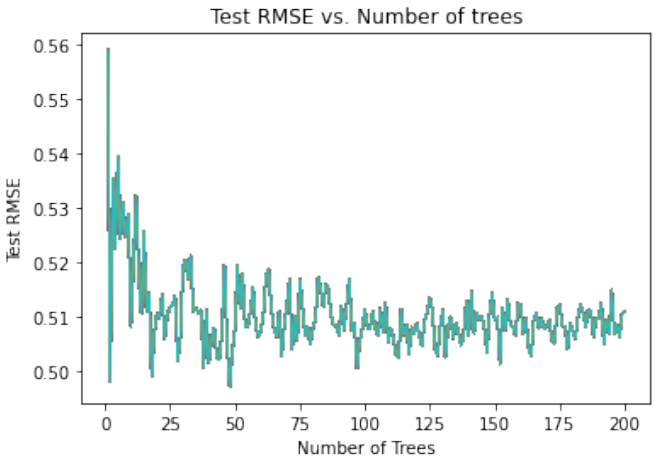



Figure 45: RMSE vs Number of Trees

From the above figure we can see that the performance initially improves with the increase of number of trees. But after some point, there is no further significant improvement in performance. This is because the larger the number of trees, the more robust aggregate model.

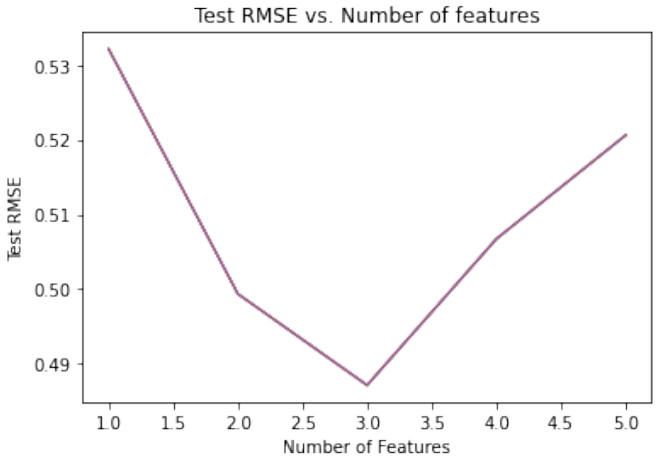



Figure 46: RMSE vs Number of Features

From the figure above we can see that increasing max features improves the performance of the model initially. But RMSE reached its minimum at max number of features of 3, the performance decreased. This is because initially by increasing the max features we have more options to consider at each node and therefore the improved performance. But after some point increasing the number of features will start decrease the diversity of individual tree and therefore hurt the performance of the model. Hence, the number of features have regularization effect.

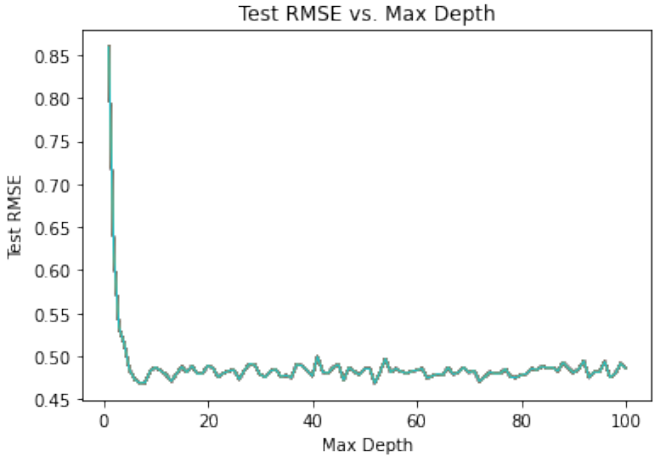



Figure 47: RMSE vs Max Depths

From the above figure we can see that the model performance increases as the max depth increases. The test RMSE reaches its minimum at max depth 7, and the model performance does not change much after this point. By increasing the depth of individual tree we increase the possible number of features we consider. The deeper the tree, the more information about the data is used in the model.

**QUESTION 22:**

Random forest is an ensemble of a large number of committees that have low correlation between each other. Random forest introduces randomness in the model by choosing from a random subset of features at each split. This will result in a low correlation between the trees and therefore diversifies the committee and results in a better model.

**QUESTION 23:**

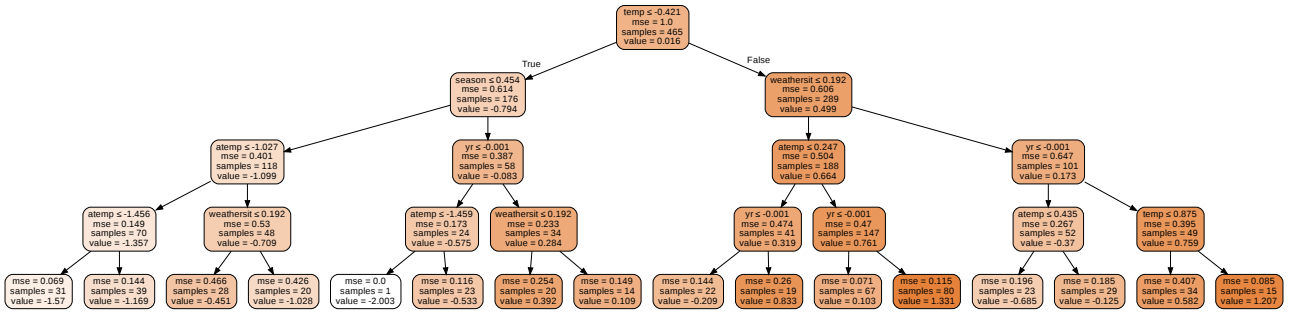



Figure 48: Tree for Bike dataset

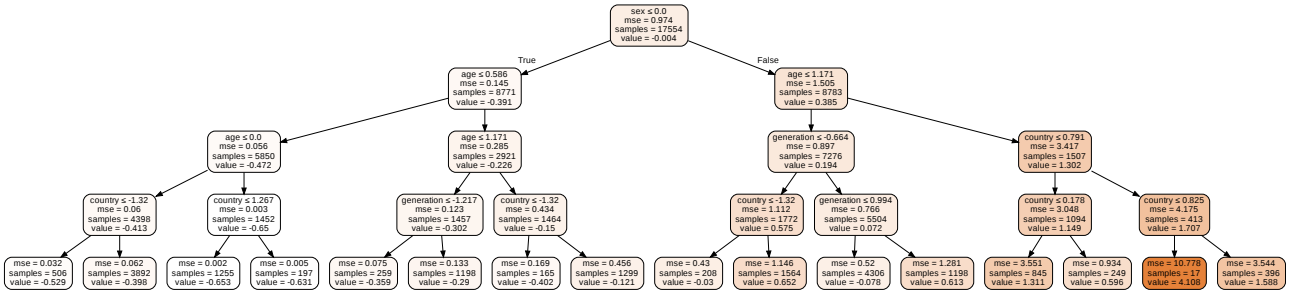



Figure 49: Tree for Suicide dataset

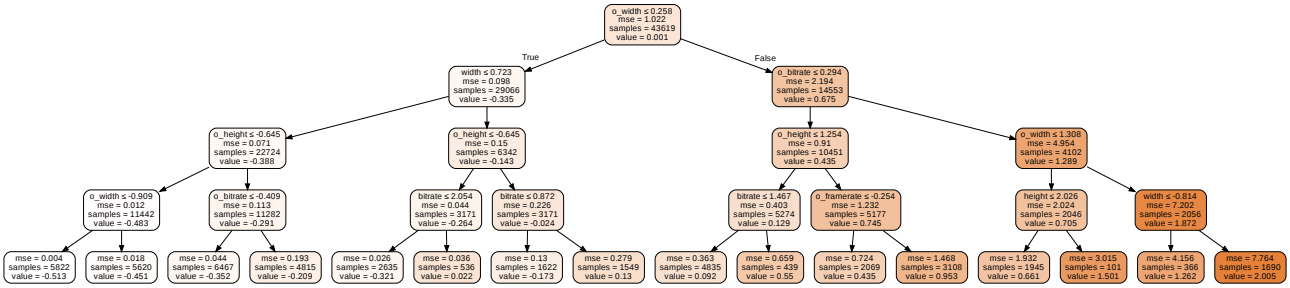



Figure 50: Tree for Video dataset

For Bike dataset, temp is selected for branching at the root node. For Suicide dataset, sex is selected for branching at the root node. For Video dataset, o-width is selected for branching at the root node. It seems that the more important the feature is for the dataset, the closer to the top of the tree. The important features match what we got in part 3.2.1 .

### 3.2.5 LightGBM, CatBoost and Bayesian Optimization

**QUESTION 24:**

To test out LightGBM and CatBoost, we select the bike dataset, and run some experiments on tunable parameters of LightGBM and CatBoost models.

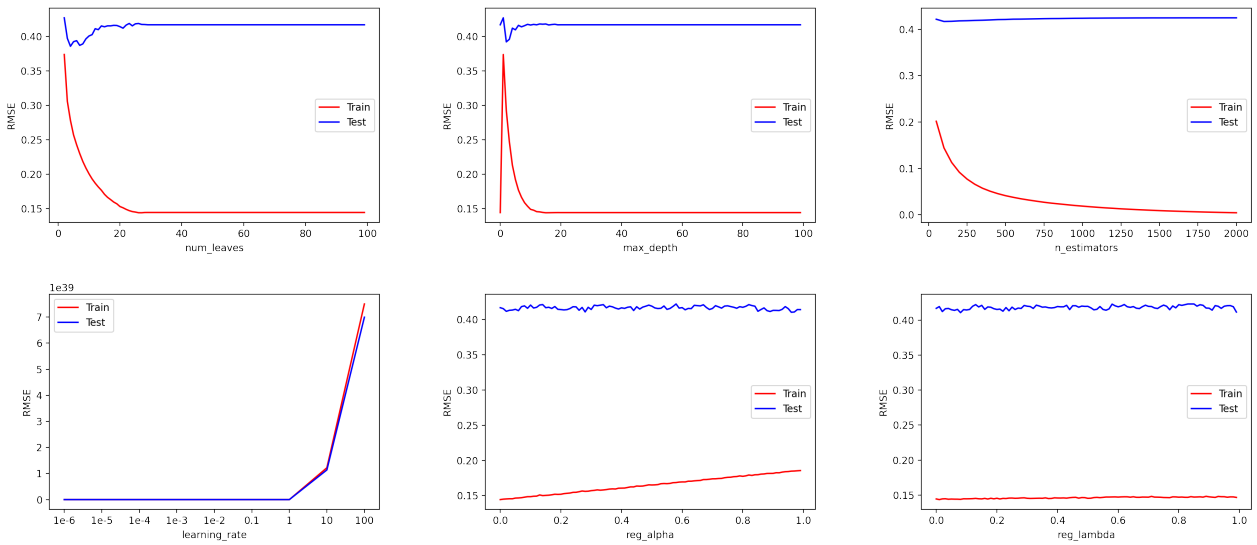



Figure 51: Experiments with parameters in LightGMB

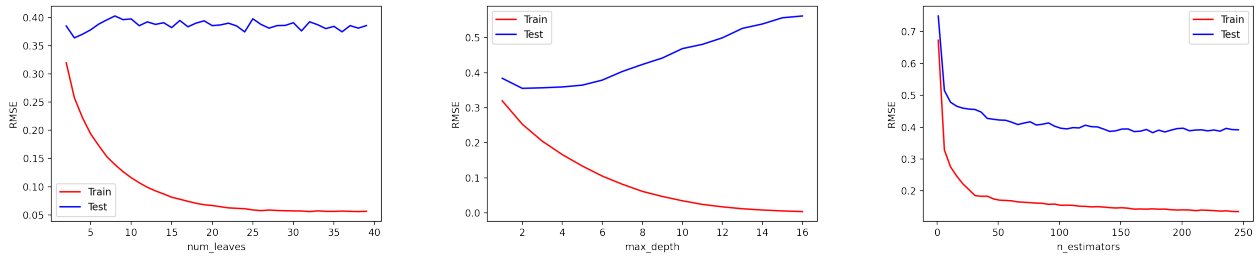



Figure 52: Experiments with parameters in CatBoost

Looking at the documentation, we see that there are several parameters we can tune for the regression model in each library. For LightGBM, it is recommended that we tune the number of leaves, number of trees, maximum depth, and the $L_2$ regularization coefficient $\lambda$. Likewise, for CatBoost, we can tune the number of trees, maximum depth, and the $L_2$ regularization coefficient.

Bayesian hyperparameter optimization will be performed on LightGBM for number of leaves from 2 to 100. For both LightGBM and CatBoost, the number of trees will be tuned in the range of 1 to 100, the depth will be tuned from 1 to 10, and the $L_2$ regularization coefficient will be tuned from $10^{-3}$ to $10^3$.

**QUESTION 25:**

For LightGBM, the best hyperparameters found are:

- number of leaves = 18

- number of trees = 89

- maximum depth = 3

- $L_2$ regularization coefficient = 7.29

Using these parameters, LightGBM achieved an average RMSE of 0.402 over 10 folds using cross validation.

For CatBoost, the best hyperparameters found are:

- number of trees = 64

- maximum depth = 8

- $L_2$ regularization coefficient = 1.83

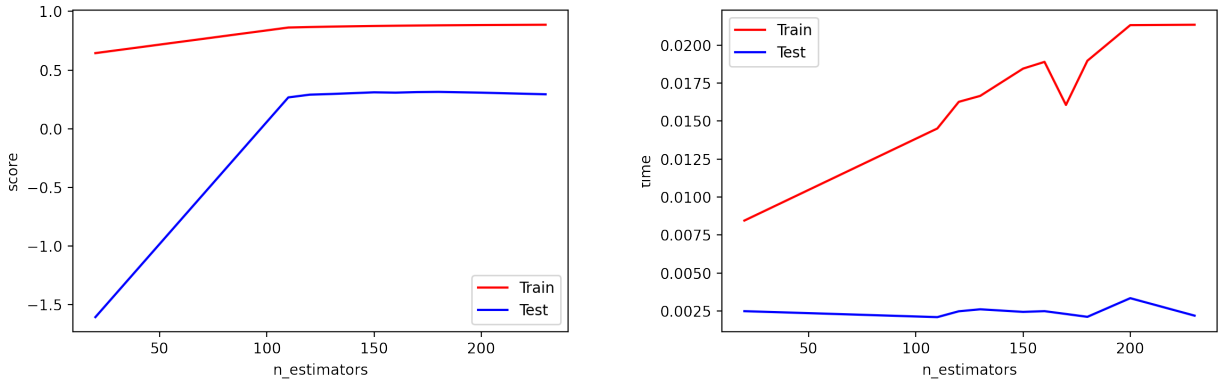
Using these parameters Light GBM achieved an average RMSE of 0.450 over 10 folds using cross validation.

**QUESTION 26:**

There are two parameters directly responsible for the performance of the models, the number of leaves and the depth of the tree. These two parameters control the capacity of a model and allow it to model functions of varying complexities. As seen in the optimal parameters discussed in Question 25 as well as in Figures 51 and 52, increasing depth and number of leaves reduces the training error.

To deal with the generalization gap, we also tuned the $L_2$ regularization coefficient. Regularization is a commonly used technique to condition ill posed problems to be amenable to optimization techniques without overfitting to the specific problem at hand. By default, both LightGBM and CatBoost do not use $L_2$ regularization. However, by using it in our Bayesian Search as discussed in Question 25, we found that we were able to get lower testing error. This provides evidence that $L_2$ regularization help to shrink the regularization gap.

The parameters most directly responsible for fitting efficiency is the number of estimators employed. Since each tree builds off the previous, it stands to reason that increasing the number of estimators will increase the training time required.




(a) score vs n_estimator

(b) time vs n_estimator

Figure 53: Influence of n_estimator on score and time

We also use correlation matrix to estimate the relationship between three main parameters {max_depth, num_leaves, n_estimators}, and the four evaluation metrics {train_score, test_scorem, fit_time, score_time}. The values for the correlation matrix is generated from the Bayesian grid search.

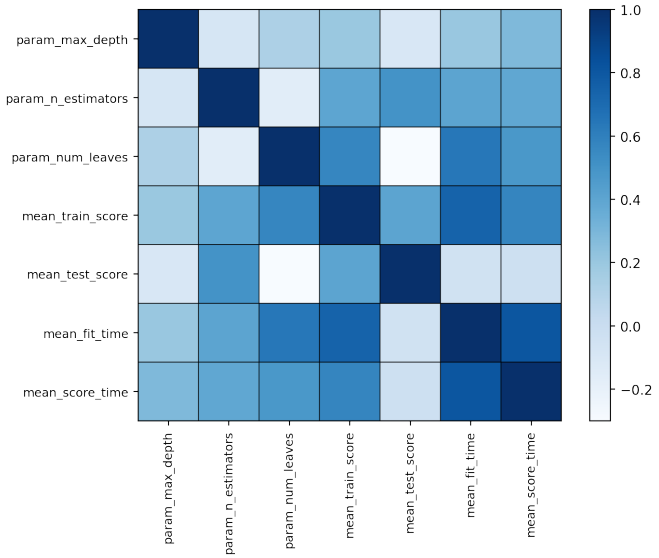



Figure 54: Correlation of parameters in LightGBM

From Fig. 54, we can see that the feature that most influence the fit time and score time in LightBoost is num_leaves. The feature most affect test score is n_estimators, and the feature most affect train score is num_leaves.

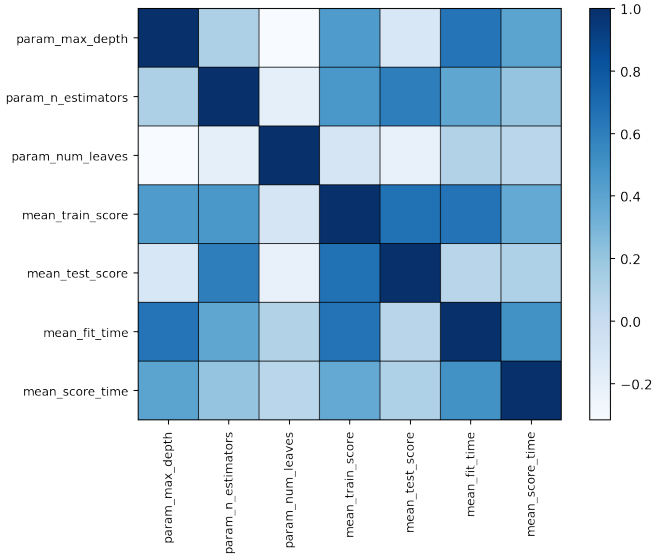



Figure 55: Correlation of parameters in CatBoost

From Fig. 55, we can see that the feature that most influence the fit time and score time in CatBoost is max_depths. The feature most affect train score and test score is n_estimators.

## 3.3 Evaluation

**QUESTION 27:**

| Dataset | Method | Parameters | train RMSE | test RMSE |
|---|---|---|---|---|
| bike | Linear | - | 0.468 | 0.502 |
| | Lasso | $\alpha = 36$ | 0.470 | 0.500 |
| | Ridge | $\alpha = 0.02$ | 0.470 | 0.495 |
| | Polynomial | Degree = 2, $k = 16$ | 0.395 | 0.485 |
| | Neural Network | Layers = (50, 100, 50) $\alpha = 0.001$ | 0.321 | 0.430 |
| | Random Forest | Trees = 1000, Depth = 90, Features = 3 | 0.150 | 0.478 |
| suicide | Linear | - | 0.894 | 0.887 |
| | Lasso | $\alpha = 0$ | 0.894 | 0.887 |
| | Ridge | $\alpha = 0.9$ | 0.894 | 0.887 |
| | Polynomial | Degree = 2, $k = 21$ | 0.852 | 0.853 |
| | Neural Network | Layers = (50, 100, 50) $\alpha = 0.001$ | 0.707 | 0.932 |
| | Random Forest | Trees = 1000, Depth = 90, Features = 3 | 0.315 | 1.021 |
| video | Linear | - | 0.816 | 0.816 |
| | Lasso | $\alpha = 0.01$ | 0.817 | 0.816 |
| | Ridge | $\alpha = 350$ | 0.817 | 0.816 |
| | Polynomial | Degree = 2, $k = 16$ | 0.782 | 0.783 |
| | Neural Network | Layers = (50, 100, 50) $\alpha = 0.001$ | 0.771 | 0.775 |
| | Random Forest | Trees = 1000, Depth = 90, Features = 3 | 0.751 | 0.799 |

Table 12: Cross Validation for Optimal Models

A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill while tuning model's hyperparameters [8].

Validation and Training would have different RMSE values as they're tested on different data and validation acts as a test of how the model is performing, before it is evaluated on the test dataset.

Generally, training underestimates the error value, so validation gives larger error value, but sometimes, validation is smaller. Possible reasons are:

1. Training was done on hard cases
2. Validation was performed on easier cases [9]

**QUESTION 28:**

R2 score provides the coefficient of determination (R**2) for the trained model on the given data. Since you pass the same data used for training, this is your overall training loss score. If you would put "unseen" test-data here, you get validation loss. [10] [11]

OOB provides the coefficient of determination using oob method, i.e. on 'unseen' out-of-bag data. This score serves as cross-validation loss and accordingly to L. Breinman oob_score = cross-validation score [10] [11]

| Dataset | Method | Parameters | Out of Bag Error |
|---|---|---|---|
| bike | Random Forest | Trees = 1000, Depth = 90, Features = 3 | 0.831 |
| suicide | Random Forest | Trees = 1000, Depth = 90, Features = 3 | 0.874 |
| video | Random Forest | Trees = 1000, Depth = 90, Features = 3 | -0.014 |

Table 13: Random Forest Out of Bag Error

# Reference

1. Linear Regression
2. Lasso regression
3. Ridge Regression
4. Neural Network
5. Random Forest
6. Feature Scaling
7. Feature Scaling - scikit learn
8. Validation
9. Validation less than Training
10. R2 and OOB - Scikit-Learn
11. R2 and OOB