Tweet Impact Prediction

1. Introduction:

1.1. Problem background:

Jahanna Chronicle, a technology company based in Lunakick, has reached out to you with a task that involves predicting the impact of a tweet and prepare a report on your analysis. You could think about 'impact' as a value that could help Jahanna Chronicle decide if the tweet could go viral.

1.2. Problem description:

Here we need to build the most efficient model, that serve the Chronicle's engineering team the best wat possible. So there are few things that must be addressed, such as:

- Your findings of data
- Most important features
- Graph of training and testing error
- Time taken by algorithms
- Overfitting and Underfitting
- Performance due
- Evaluation of algorithms
- Best performing algorithm and why?
- Formula from your regression algorithm
- Evaluation of formula

2. Data Section:

2.1. Dataset preparation:

Chronicle's data team has worked hard and prepared a dataset for you. They compiled the data and decided to share it via HTTP. So here is the link, directly from the Chronicle's data team!

2.2. Data description:

There are 15 features and 1 dependent variable (also called as the output variable; Here it is named as 'impact').

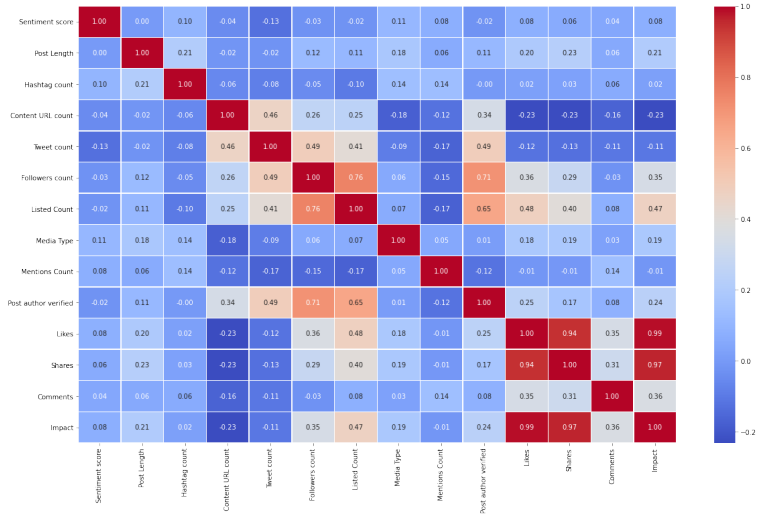
1. Post Content - The text in the tweet
2. Sentiment score - Ranges from -20 to +20 (0 - neutral)
3. Post Length - The length of the tweet
4. Hashtag Count - The number of hashtags used in the tweet
5. Content URL Count - The number of URLs mentioned in the tweet
6. Tweet Count - The total number of tweets posted by the author of the tweet
7. Followers Count - The number of followers of the author of the post
8. Listed Count - the number of lists the post author is a part of
9. Media Type - The media type of the post (Text, image, video)
10. Published Datetime - The published time of the tweet
11. Mentions Count - The number of user mentions in the tweet
12. Post Author Verified - 1 if author is a verified user
13. Likes - Likes received for the tweet
14. Shares - Retweets received for the tweet
15. Comments - Number of comments for the tweet

3. Methodology:

3.1. Data analysis:

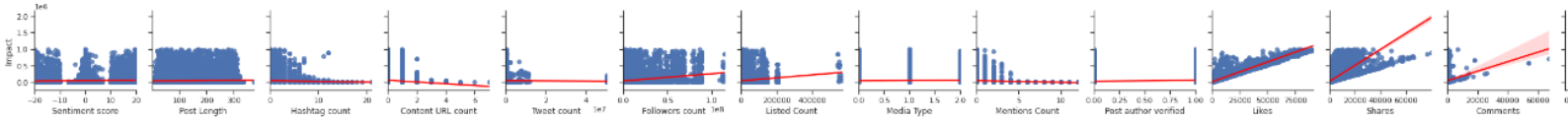
3.1.1.1. **Missing values:** There are zero missing values in the dataset

3.1.1.2. **Correlation:**



From the correlation analysis highly correlated features are **Likes** and **Shares**.

3.1.1.3. **Linearity of feature:**



From the above graphs, we can see that Likes, Shares, and comments show a linear relationship with Impact. So if we train a model with these three features only, then linear regression will give the best result

3.1.1.4. **Importance of feature (using R squared and RMSE):**

Why adjusted R square?

The adjusted R-squared compares the explanatory power of regression models that contain different numbers of predictors. ... The adjusted R-squared increases only if the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected by chance

feature	adj_r2_diff	r2_wo_feature	r2_with_feature	RMSE_without	RMSE_with
Likes	3.986389e-01	0.601369	1.0	64847.763639	3.467314
Shares	6.369230e-02	0.936309	1.0	26774.320567	3.467314
Comments	2.774390e-03	0.997226	1.0	2235.642262	3.467314
Listed Count	1.118994e-12	1.000000	1.0	3.467609	3.467314
Post author verified	7.106538e-13	1.000000	1.0	3.467255	3.467314
Followers count	4.762857e-13	1.000000	1.0	3.467576	3.467314
Content URL count	4.292122e-13	1.000000	1.0	3.467202	3.467314
Post Length	2.469136e-13	1.000000	1.0	3.467342	3.467314
Hashtag count	8.548717e-14	1.000000	1.0	3.467472	3.467314
Sentiment score	4.529710e-14	1.000000	1.0	3.467353	3.467314
Media Type	2.176037e-14	1.000000	1.0	3.467129	3.467314
Mentions Count	1.232348e-14	1.000000	1.0	3.467313	3.467314
Tweet count	0.000000e+00	1.000000	1.0	3.467314	3.467314

Adjusted r square values decrease and RMSE value increases without likes, share and comments, and remains almost the same without other features. So likes, share and comments are enough to predict the impact, we call this hypothesis is hypo_1

3.1.1.5. Comparing different algorithms on whole feature set:

```
LR: 49.167461
[ 16.26647733   1.13673868  59.75710356  11.99767649  24.33525139
 241.1410418   15.52730894   4.00160807   2.94259909 114.56880166]

KNN: 9166781.735840
[ 6376061.6152  4193882.652  30382339.788  20497878.8904 12611475.572
  2625035.0496  2580253.2152  2835022.8208  4106717.1392  5459150.616 ]

DT: 8654002.254232
[ 7118847.32051057  9739735.25801646  8853756.31966067 11428894.75763694
  7411699.45817888  7009099.39734714 10297891.00732714  6301651.66479023
  9603370.89666831  8775076.46218201]

RF: 6088711.771905
[ 3538421.61728146  5422254.51420088 10766445.4899106  11589995.67995915
  8155037.10332362  3294412.31051274  3377292.95870122  2978160.54930848
  6459474.77279159  5305622.72306359]

EXT: 3760516.422931
[  434498.70382377  3393735.67771296  8806898.53259364 10394397.72152837
  4029908.69442     1365660.60365676  1190838.94640914  1249526.09024851
  2577332.33206531  4162366.92685434]

GBM: 11962141.024981
[11393165.16814155 13499701.73923666 15819975.72844438 10948104.11022444
 13333806.49135623 13470755.15308519 11866957.31563984  9991614.20637445
 10053952.12844927  9243378.20885567]

XGB: 13101640.854520
[11169648.76916186 14911650.09693147 17121657.40179495 13059503.92727088
 15063672.72948988 14501061.45122977 12620218.71721752 10301283.71095573
 11238748.27316439 11028963.46797977]

LGBM: 44495517.214571
[36555180.52524768 39612021.64236668 73267823.64661677 66634968.39087052
 49320761.94709057 39443245.72245554 42157767.96244963 32843395.64124986
 40702304.41472893 24417702.2526321 ]

NN: 5495.155452
[1.02799195e+02 1.65894166e+03 1.20920409e+03 4.81113543e+04
 2.34989809e+03 2.83176418e+02 4.52193871e+01 2.94957408e+02
 4.64932157e+01 8.49510809e+02]
```

Linear model gives the lowest mean squared error result as compare to tree base models and neural networks

3.1.1.6. Comparing different algorithms on hypo_1(Likes, Shares, Comments):

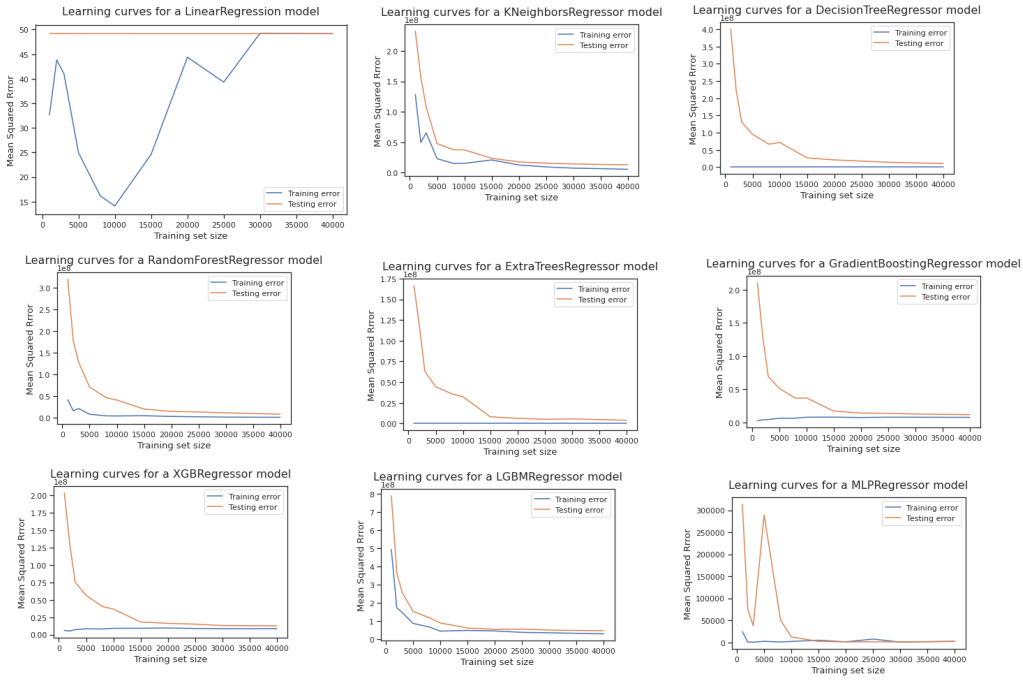

```
LR: 49.167461
[ 16.26647733   1.13673868  59.75710356  11.99767649  24.33525139
 241.1410418   15.52730894   4.00160807   2.94259909 114.56880166]

KNN: 9166781.735840
[ 6376061.6152  4193882.652  30382339.788  20497878.8904 12611475.572
  2625035.0496  2580253.2152  2835022.8208  4106717.1392  5459150.616 ]

DT: 8654002.254232
[ 7118847.32051057  9739735.25801646  8853756.31966067 11428894.75763694
  7411699.45817888  7009099.39734714 10297891.00732714  6301651.66479023
  9603370.89666831  8775076.46218201]

RF: 6088711.771905
[ 3538421.61728146  5422254.51420088 10766445.4899106  11589995.67995915
  8155037.10332362  3294412.31051274  3377292.95870122  2978160.54930848
  6459474.77279159  5305622.72306359]

EXT: 3760516.422931
[  434498.70382377  3393735.67771296  8806898.53259364 10394397.72152837
  4029908.69442     1365660.60365676  1190838.94640914  1249526.09024851
  2577332.33206531  4162366.92685434]

GBM: 11962141.024981
[11393165.16814155 13499701.73923666 15819975.72844438 10948104.11022444
 13333806.49135623 13470755.15308519 11866957.31563984  9991614.20637445
 10053952.12844927  9243378.20885567]

XGB: 13101640.854520
[11169648.76916186 14911650.09693147 17121657.40179495 13059503.92727088
 15063672.72948988 14501061.45122977 12620218.71721752 10301283.71095573
 11238748.27316439 11028963.46797977]

LGBM: 44495517.214571
[36555180.52524768 39612021.64236668 73267823.64661677 66634968.39087052
 49320761.94709057 39443245.72245554 42157767.96244963 32843395.64124986
 40702304.41472893 24417702.2526321 ]

NN: 5495.155452
[1.02799195e+02 1.65894166e+03 1.20920409e+03 4.81113543e+04
 2.34989809e+03 2.83176418e+02 4.52193871e+01 2.94957408e+02
 4.64932157e+01 8.49510809e+02]
```


Result from hypo_1:

1. Linear_model gives the best result i.e., Linear Regression
2. Tree base models like decision tree, bagging(Random forest and Extra trees), and Bagging (GradientBoosting, XGBoost, and LightGBM) give extremely high errors
3. Neural network (Multi-layer Perceptron) performance lies between Linear and Tree base models, but as compare to Linear Regression, Neural network error is a very high error

The result from hypo_1 is better than the whole because here we only consider the three features which improve performance and efficiency as the error difference is only 0.02 for Linear models, which is negligible.

3.1.1.7. Learning curves(Graph of training and testing error):



Except for Linear regression, the Mean squared error for other algorithms are lies between 5495 and 44495517. Due to 10 to the power 8(1e8) conversation of Mean squared error on Y-axis, the gap between training and testing error looks very small on graph but in reality, it is very high as compared to Linear regression.

Linear model (Linear regression):

1. Range of Mean squared error for Linear regression is 12 to 50 Training and testing error marges at (Mean squared error =~ 50 and training set size =~ 30000). Adding more training instances till 30000 is very likely to lead to better models under the current algorithm but after 30000 training instances won't increase the model performance.
2. From the graph we can see that when the training size was less than 30000 model suffer from overfitting because the testing error was more than training error but after 30000 testing error =~ training error, so we can conclude that to avoid overfitting, train set size >= 30000
3. Underfitting is when the training error is high but in our case, there are no signs of Underfitting
4. Linear model gives testing error =~ training error, this indicates that linear model is the best fit solution for this problem

3.1.1.8. Comparing different linear models:

```
LR: 49.167461
[ 16.26647733   1.13673868  59.75710356  11.99767649  24.33525139
 241.1410418   15.52730894   4.00160807   2.94259909 114.56880166]
training time: 0.136 s

BR: 49.167461
[ 16.26647733   1.13673868  59.75710356  11.99767648  24.3352514
 241.1410418   15.52730894   4.00160807   2.94259909 114.56880166]
training time: 0.245 s
```

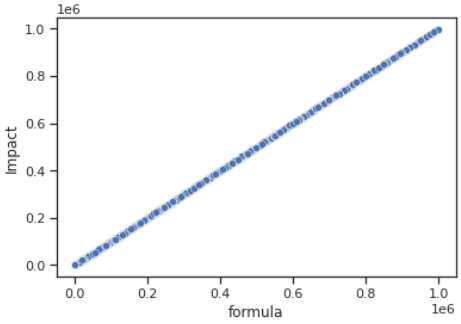
Linear Regression and BayesianRidge give the same result but Linear Regression almost 1.65 times faster than BayesianRidge So we will go with **Linear Regression**

Note : Time taken by the whole dataset is 0.035sec and by hypo_1 is 0.010. hypo_1 is 3.5 times faster and gives the same results.

3.1.1.9. Final reuslt from Linear Regression:

1. training time = 0.005 s and testing time = 0.005 s
2. Coefficients = [[10.00000502 10.0000079 10.00000524]]
3. Root Mean Squared Error (RMSE) = 3.467
4. Score = 0.9999
5. Coefficient of determination(R square) = 1.00

3.1.1.10. Formula (created using Coefficients of Linear Regression):



1. Impact = (10 x Likes) + (10 x Shares) + (10 x Comments)
2. time required in applying the formula on whole dataset: 0.006 s, which is 1.67 times faster than Linear Regression
3. Graph between formula and Impact, gives single straight line, So the formula is correct.
4. Coefficient of determination(R square) using the formula: 1.00
5. Acurracy score using the formula 0.99802

4. Conclusion:

1. No missing values
2. Likes and share are highly correlated with Impact
3. Likes, Shares, and comments show the linear relationship with Impact
4. Adjusted r square values decrease without likes, share, and comments, and remains almost the same without other features. So likes, share, and comments are enough to predict the impact, we call this hypothesis is hypo_1
5. Linear model gives the lowest mean squared error result as compare to tree base models and neural networks
6. Result from hypo_1:
    - The Linear model gives the lowest mean squared error result as compare to tree base models and neural networks
    - Result from hypo_1 is better than the whole because here we only consider the three features which improve performance and efficiency as error difference are only 0.02 for linear models, which is negligible.
    - Time taken by the whole dataset is 0.035sec and by hypo_1 is 0.010.hypo_1 is 3.5 times faster and gives the same results

****Note: From the above result, all following analyses will be based on hypo_1****

7. Analysis from learning curves (training vs testing):
    - Except for Linear regression, the Mean squared error for other algorithms are lies between 5495 and 44495517. Due to 10 to the power 8(1e8) conversation of Mean squared error on Y-axis, the gap between training and testing error looks very small on graph but in reality, it is very high as compared to Linear regression.
    - Range of Mean squared error for Linear regression is 12 to 50 Training and testing error merges at (Mean squared error =~ 50 and training set size =~ 30000). Adding more training instances till 30000 is very likely to lead to better models under the current algorithm but after 30000 training instances won't increase the model performance.
    - From the above graph, when the training size was less than 30000 model suffer from overfitting because testing the error was more than training error but after 30000 testing error =~ training error, so we can conclude that to avoid overfitting, train set size >= 30000
    - Underfitting is when the training error is high but in our case, there are no signs of Underfitting
    - Linear model gives testing error =~ training error, therefore the linear model is the best fit solution for this problem

****Note: From the above result, all following analyses will be based on hypo_1 and linear models****

8. Comparing linear models: Linear Regression and BayesianRidge give the same result but Linear Regression is almost 1.65 times faster than BayesianRidge. So we will go with Linear Regression
9. Final result from Linear Regression:
    - training time = 0.005 s and testing time = 0.005
    - Coefficients = [[10.00000502 10.0000079 10.00000524]]
    - Root Mean Squared Error (RMSE) = 3.467
    - Score = 0.9999
    - Coefficient of determination(R square) = 1.00

10. Result from the formula created using Coefficients of Linear Regression:
    - Formula: Impact = (10 x Likes) + (10 x Shares) + (10 x Comments)
    - time required in applying the formula on whole dataset: 0.006 s, which is 1.67 times faster than Linear Regression
    - Graph between formula and Impact, gives single straight line, so the formula is correct.
    - Accuracy score using the formula 0.99802
    - Coefficient of determination(R square) using the formula: 1.00