2024

Report

ML Project Report

Team: SC_33

1. عبدالرحمن حمدى عبدالتواب احمد
   20201701096
2. عبدالرحمن اسامه عبدالكريم محمود
   20201701088
3. عبدالرحمن مجدي أحمد رزق
   2021170302
4. حبيبه احمد عبدالقادر السيد
   20201700218
5. زياد احمد السيد حسن ابراهيم
   20201700299

Phase 1

Preprocessing:

1- We need to remove the outliers from the dataset so it doesn't affect on the model ,and replaced the outliers with the mean of each column like in the Pic.

```
def detect_outliers_z_score(data):
    columns_to_check = [col for col in DF.columns
                        if col not in ['url', 'title', 'channel type', 'weekday', 'isWeekEnd']]
    outliers = {}
    threshold = 3
    for col in data.columns:
        if col in columns_to_check:
            z_scores = np.abs(stats.zscore(data[col]))
            outliers[col] = data.index[z_scores > threshold].tolist()
    return outliers


outliers = detect_outliers_z_score(DF)
for col, indices in outliers.items():
    for index in indices:
        DF.at[index, col] = DF[col].mean()
```

2- We applied scaling(Min-Max) factor to all the data so we don't have any zero's or negative values in the data set and to increase the model performance.

```
columns_to_scale = [col for col in DF.columns
                    if col not in ['url', 'title', 'channel type', 'weekday', 'isWeekEnd']]

scaler = MinMaxScaler()

DF[columns_to_scale] = scaler.fit_transform(DF[columns_to_scale])
```

3- We mapped the columns with string statements to float values so all the data set will be numerical Because the model can't Handle the string data.

```
channel_type_mapping = {
    ' data_channel_is_bus': 0.0,
    ' data_channel_is_socmed': 1.0,
    ' data_channel_is_lifestyle': 2.0,
    ' data_channel_is_world': 3.0,
    ' data_channel_is_entertainment': 4.0,
    ' data_channel_is_tech': 5.0,
    '[]': 6.0
}
weekday_mapping = {
    'monday': 0.0,
    'tuesday': 1.0,
    'wednesday': 2.0,
    'thursday': 3.0,
    'friday': 4.0,
    'saturday': 5.0,
    'sunday': 6.0
}
isWeekEnd = {
    'No': 0.0,
    'Yes': 1.0
}
```

```
DF['weekday'] = DF['weekday'].map(weekday_mapping)
DF['channel type'] = DF['channel type'].map(channel_type_mapping)
DF['isWeekEnd'] = DF['isWeekEnd'].map(isWeekEnd)
```

Analysis

- after performing some technique on the data the data hadn’t any **NULL** values .
- by apply correlation on the data some columns has some relationship and some are not but the relationship with the target “shares” not that powerful because correlation did not reach 0.3 .
- so , we use all the data except 3 columns “Title , URL, timedelta” because the first two will not really be useful because unique string and didn’t treat with string and third when we drop it , the mse has be better .

```
DF = DF.drop(columns=['title', 'url', ' timedelta'])
corr = DF.corr()
top_feature = corr.index[abs(corr[' shares']) > 0.1]
top_corr = DF[top_feature].corr()
```

Models

- we use two model regression one linear regression and the other polynomial regression

```
# Linear Regression Model
sln = linear_model.LinearRegression()
sln.fit(X_train, Y_train)
y_predicted = sln.predict(X_train)
prediction = sln.predict(X_test)
```

```
poly_features = PolynomialFeatures(degree=2)
X_train_poly = poly_features.fit_transform(X_train)
poly_model = linear_model.LinearRegression()
poly_model.fit(X_train_poly, Y_train)
y_train_predicted = poly_model.predict(X_train_poly)
y_test_predicted = poly_model.predict(poly_features.transform(X_test))
```

- the difference between the two is when applying polynomial regression the difference was not bad at all with polynomial regression of degree 2**.**
- we didn't increase the degree because overfitting and we need a much more powerful processor like **server** for this much data.
- we depend on the correlation of the data with the target and also Wrapper Methods for decided what feature the negative effect on the models .
- with also test size 10% from the data and 90% for train and the test was small but it make a better mse and we didn't use validation.

```
--- Linear Regression ---
Mean Square Error Train Model 1 :  2.0431010208321535e-05
Mean Square Error Test Model 1 :  1.5960432544826525e-06
--- Polynomial Regression ---
Mean Square Error Train Model 2 :  1.936162398298325e-05
Mean Square Error Test Model 2 :  6.777349922801626e-07

Process finished with exit code 0
```

Conclusion

This phase has disappointment us because really bad data and with mse such bad with regression and this data has no regression look ever .

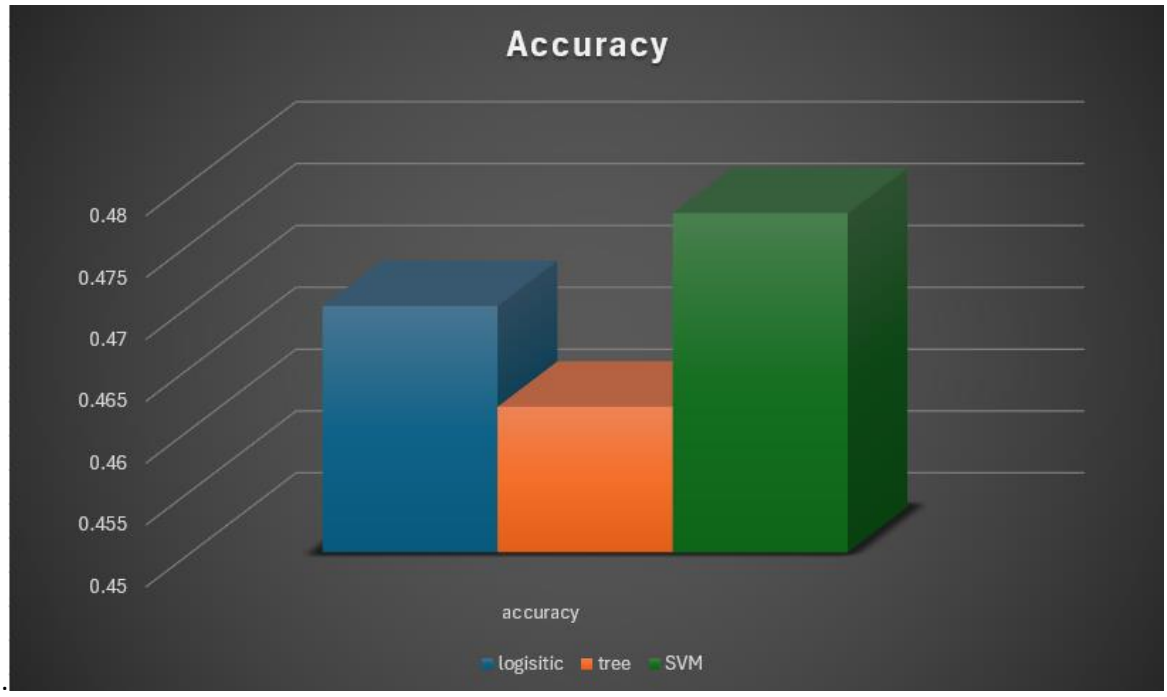
Phase 2

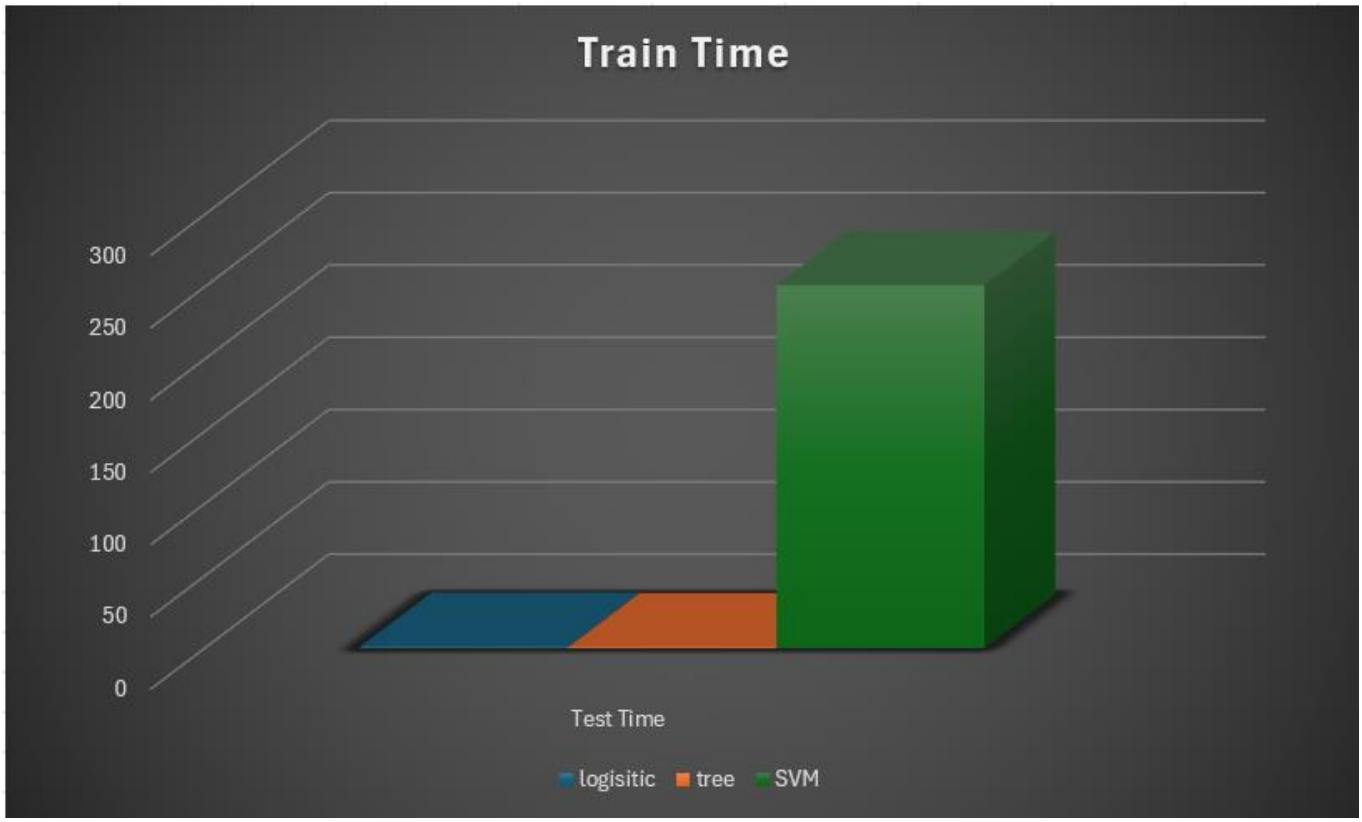
- **Preprocessing**
  - In this phase , we make some small changes like
    - Replace null values with mean of the column
    - Add column ‘Time delta’ to enhance the accuracy
    - We do preprocessing after split the data to train and test

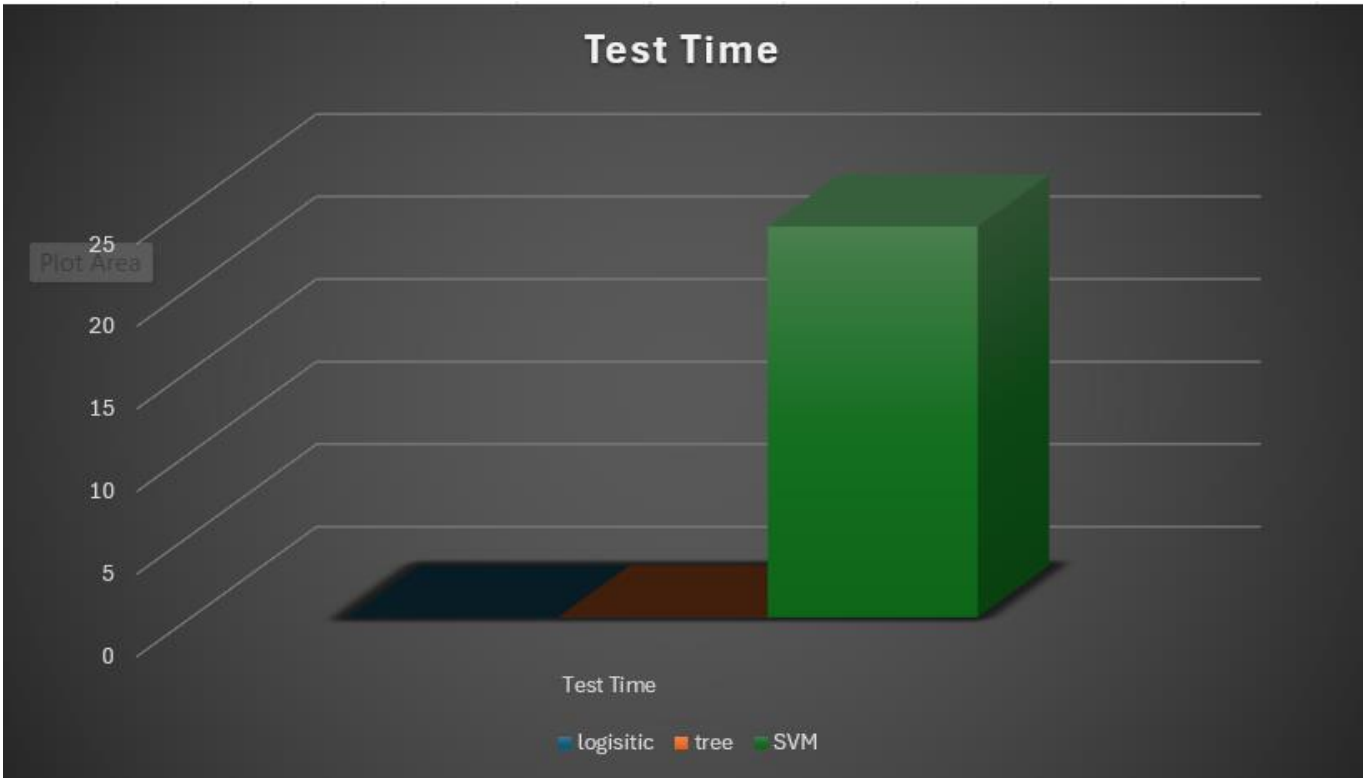
- **Analysis**

  Nothing change about the last Phase

- **Result**




Train Time
300
250
200
150
100
50
0
Test Time
logisitic
tree
SVM


Test Time
Plot Area
25
20
15
10
5
0
Test Time
logisitic
tree
SVM

- **Parameter Tuning**

we have three models

1. Logistic Regression

The two parameters are C and Max_iter

1. C = 0.5 , max iter = 1000
   Train Accuracy: 0.466584718897586 85
   Test Accuracy: 0.4692715745892095

2. C = 1 ,max iter = 1000
   Train Accuracy : 0.4662612408617455
   Test Accuracy : 0.469918488808384

3. C = 1.5, max iter = 1000
   Train Accuracy : 0.4659054150223199 7
   Test Accuracy : 0.4700478716522189

4. C = 1.5 , max iter = 100
   Train Accuracy : 0.4670052403441806
   Test Accuracy : 0.4701772544960538

5. C = 1.5 , max iter = 10000
   Train Accuracy : 0.4659054150223199 7
   Test Accuracy : 0.4700478716522189

6. C = 1.5 , max iter = 100000
   Train Accuracy : 0.4659054150223199 7
   Test Accuracy : 0.4700478716522189

2. Decision Tree

We have two parameters to tuning max depth, min sample split

1. Depth = 10, sample = 10
   Test Accuracy : 0.4468883426057705
2. Depth = 10, sample = 2
   Test Accuracy : 0.4477940225126148 5
3. Depth = 10, sample = 5
   Test Accuracy : 0.447276491137275 2
4. Depth = 5, sample = 2
   Test Accuracy : 0.4617673696467848 3
5. Depth = 20, sample = 2
   Test Accuracy : 0.3978522447923405 5
6. Depth = 2, sample = 2
   Test Accuracy : 0.435243886660628 8

3. SVM

We have two parameters kernal and C

1. C = 1, kernal = rbf
   Test Accuracy : 0.4774226937508086 5
2. C = 1.5, kernal = rbf
   Test Accuracy : 0.478328373657653
3. C = 0.5, kernal = rbf

Test Accuracy : 0.4730236770604218

4. C = 1, kernal = linear

   Test Accuracy : 0.4659076206495018 6

5. C = 1, kernal = poly with degree 2

   Test Accuracy : 0.4673308319316858 4

6. C = 1, kernal = sigmoid

   Test Accuracy : 0.3537326950446370 6

- **Conclusion**

  The classification in this this phase with the updated data has a point on the regression specific the linear regression but it not going far from polynomial that if had a big degree can be close enough ,finally the problem with ML not achieving the required accuracy even with more ml classifier like Random Forest which achieve about 0.48 is test and 1.0 in train still this far from using more powerful and big deep learning model , and thank you for your attention.