Video Game Sales on Steam Prediction

Phase 2 | Team ID: CS_8

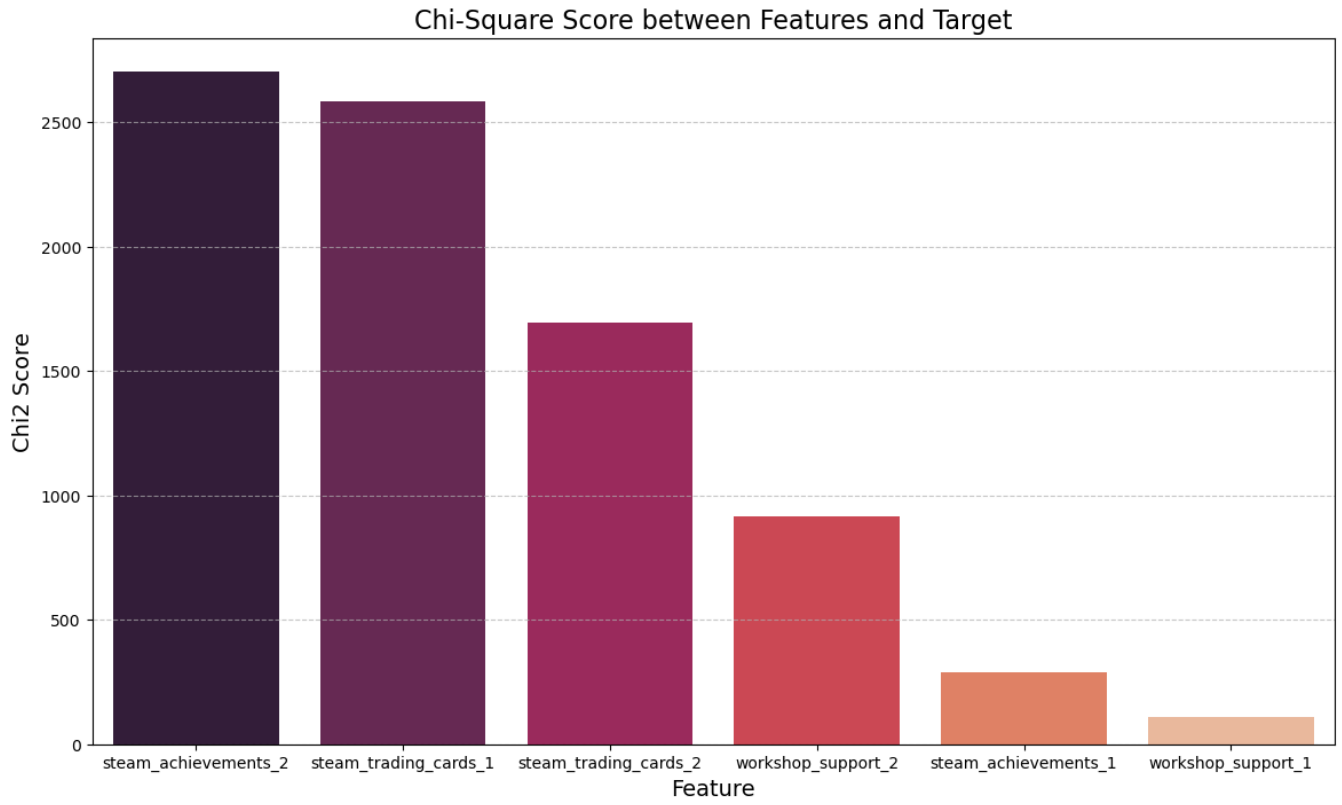
ID	Name
2022170385	محمد متولي عبدالحميد عوض محمد
2022170375	محمد عادل علي حسن
2022170389	محمد منير تاج الدين منصور
2022170373	محمد طارق الحسين محمد منصور العراقي
2022170456	مينا باسم نادي

Table of Contents

Feature Selection

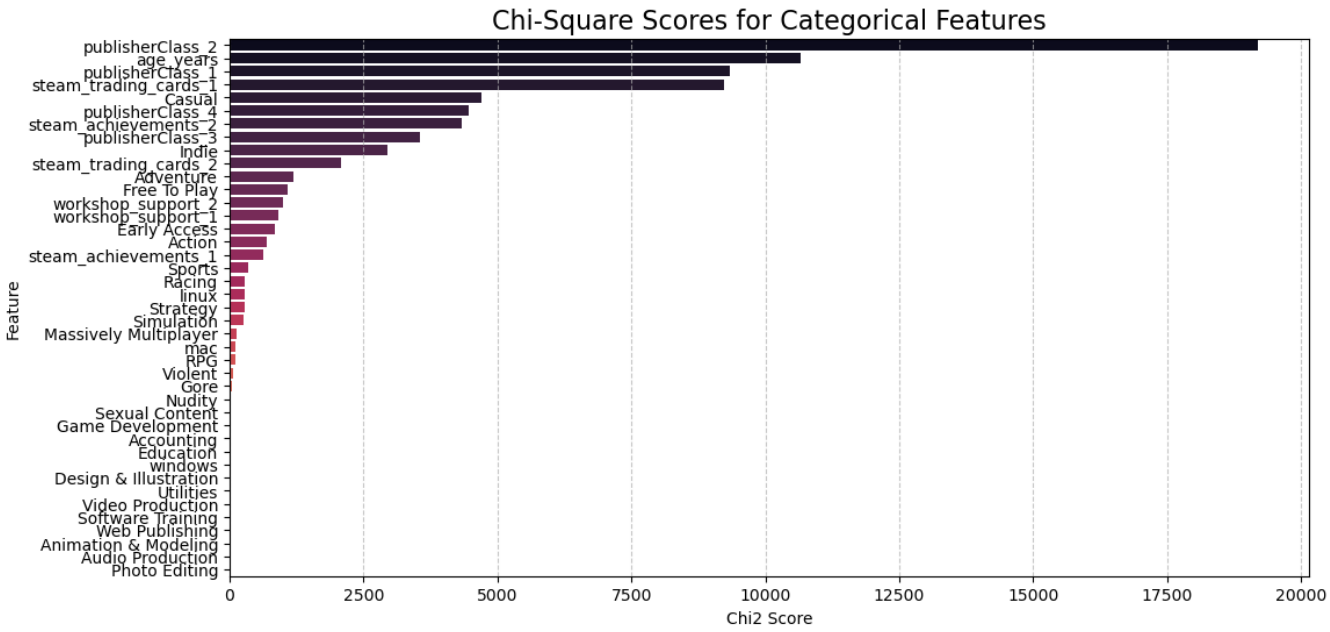
Binary features:

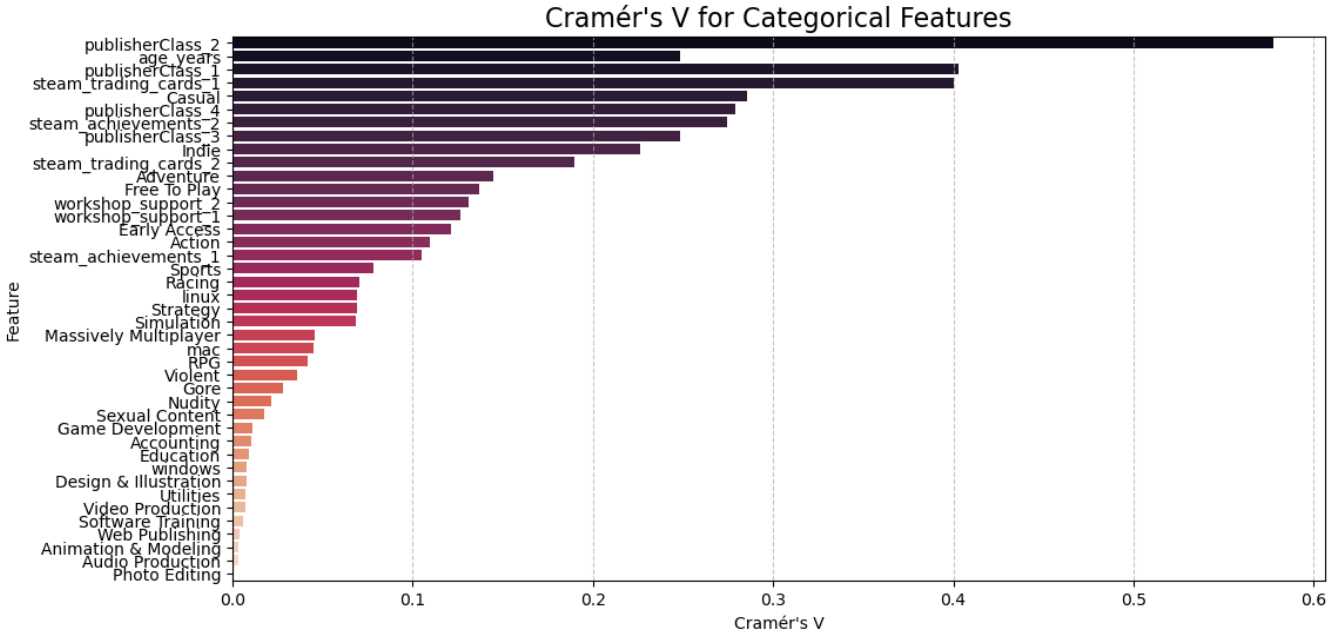
- using **Chi-square**.
- all features had **good Chi-square** scores.
- **removed no features**.



Categorical features:

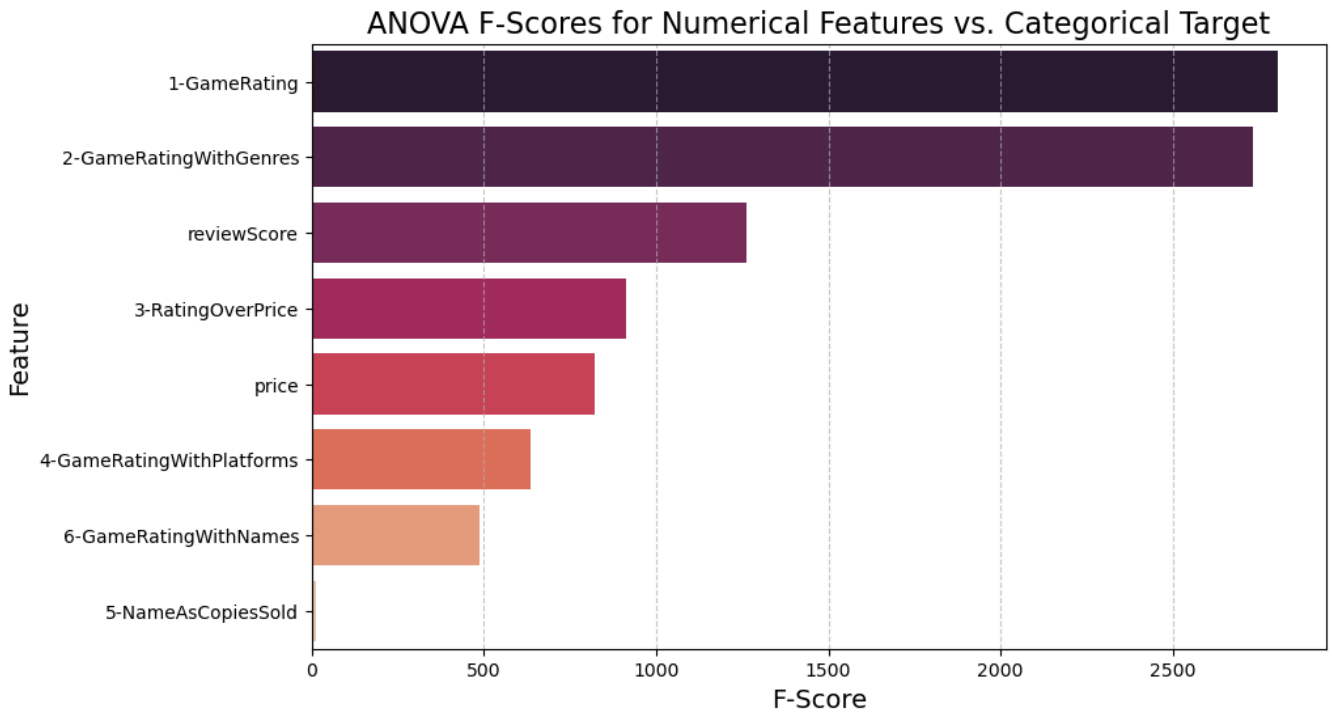
- using **Chi-square** and **Cramér's V**.
- **low** Chi-square and Cramér's V scores **made a difference** in evaluation.
- **removed features** with (**Chi-square < 10**) and (**Cramér's V < 0.04**)





Continuous features:

- using **ANOVA** (**like MS1**).
- **low** F-score features made **no difference**.
- **removed no features**.

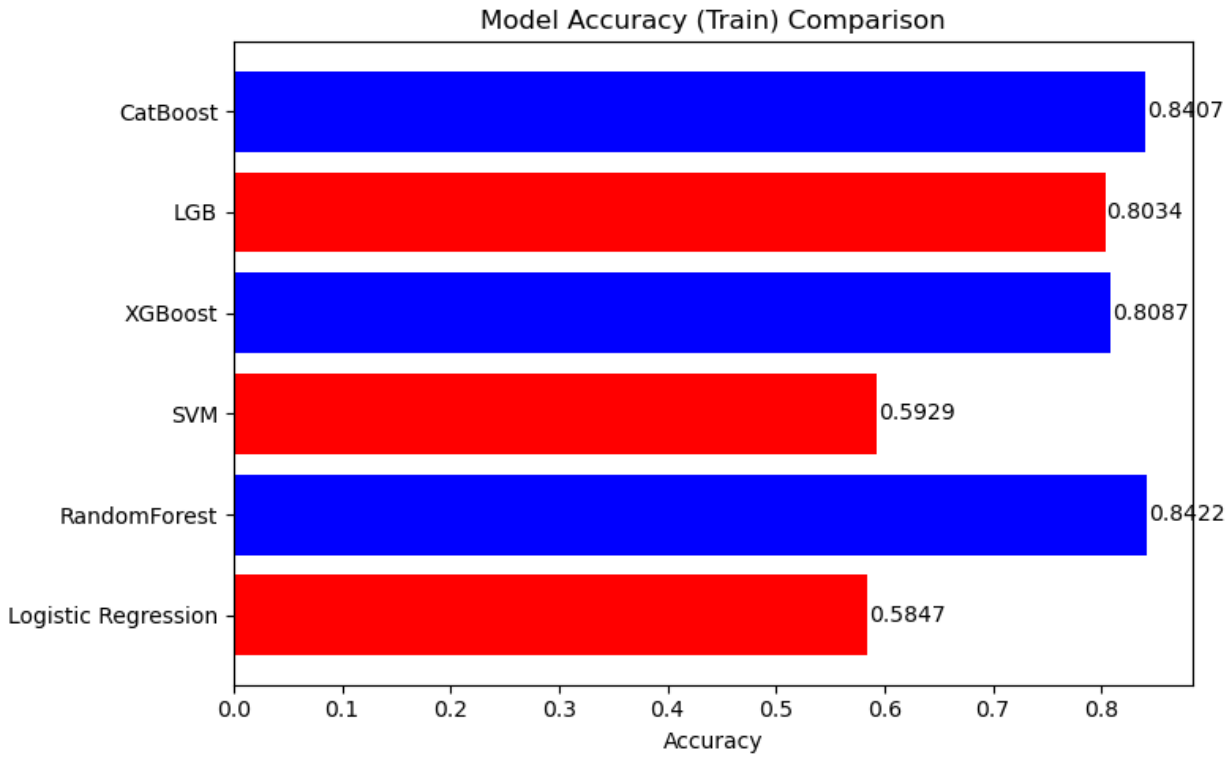


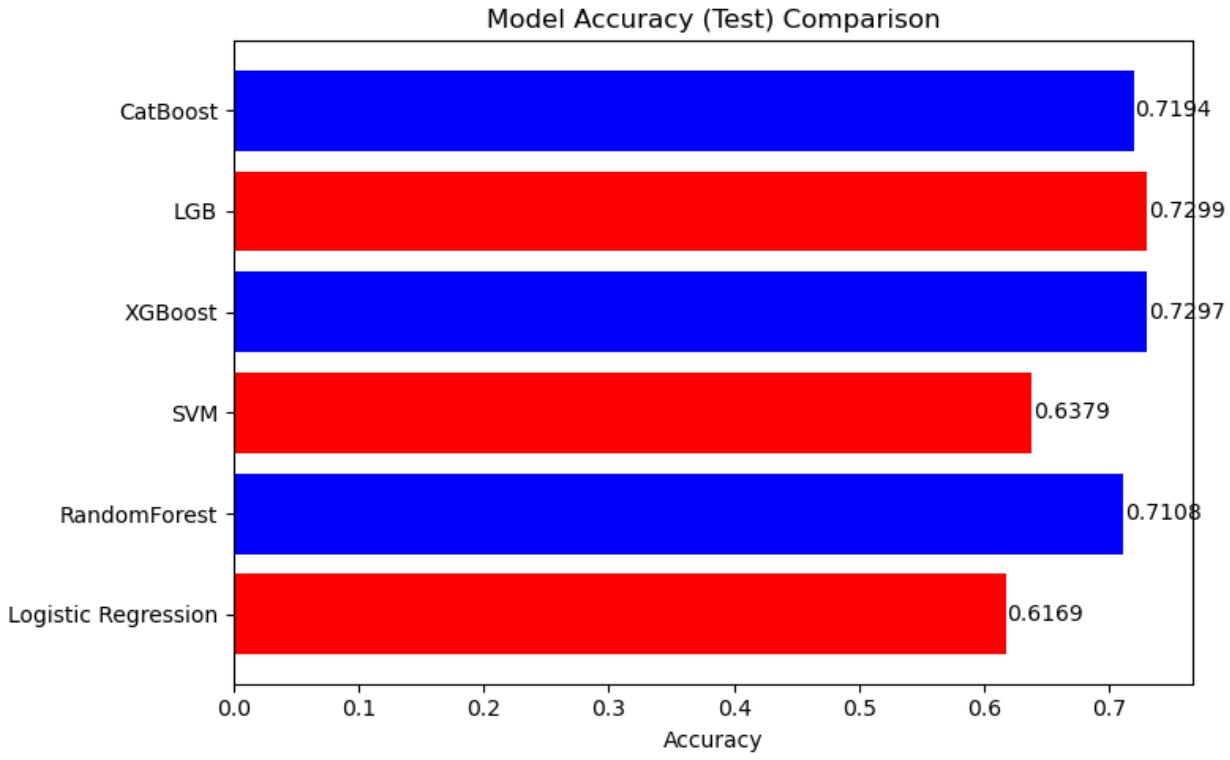
Hyperparameter Tuning

After manually tuning hyperparameters and finishing it up with GridSearch, the best hyperparamter values were:

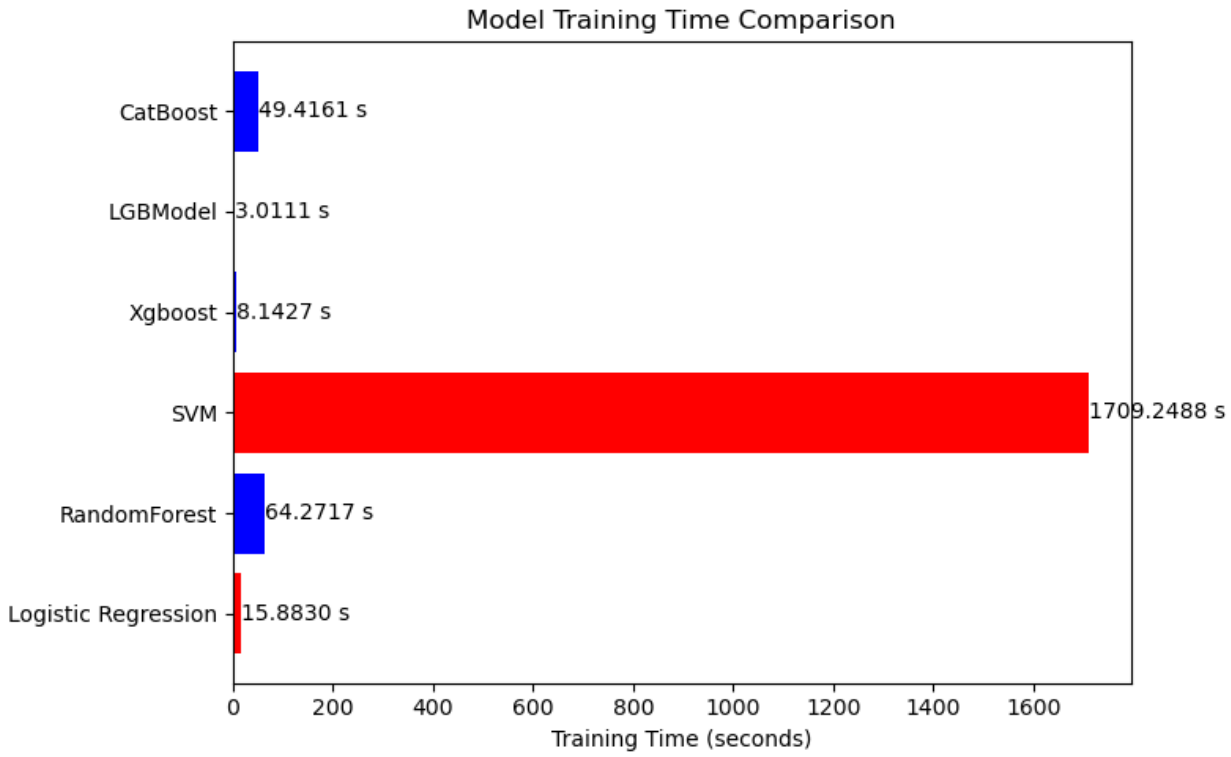
- **Logistic:** (C=10, penalty='l2', solver='lbfgs', max_iter=1000)
- **RandomForest:** (max_depth=20, max_features='sqrt', min_samples_leaf=3, min_samples_split=15, n_estimators=600, random_state=42)
- **SVM:** (C=5, kernel='rbf', gamma='scale', probability=True)
- **XGBoost:** (n_estimators=470, learning_rate=0.11, max_depth=5, subsample=0.9, colsample_bytree=0.8, random_state=42, n_jobs=-1)
- **LGBM:** (learning_rate=0.06, max_depth=15, n_estimators=200, num_leaves=50)
- **CatBoost:** (iterations=350, learning_rate=0.1, depth=10, verbose=0, random_state=42)

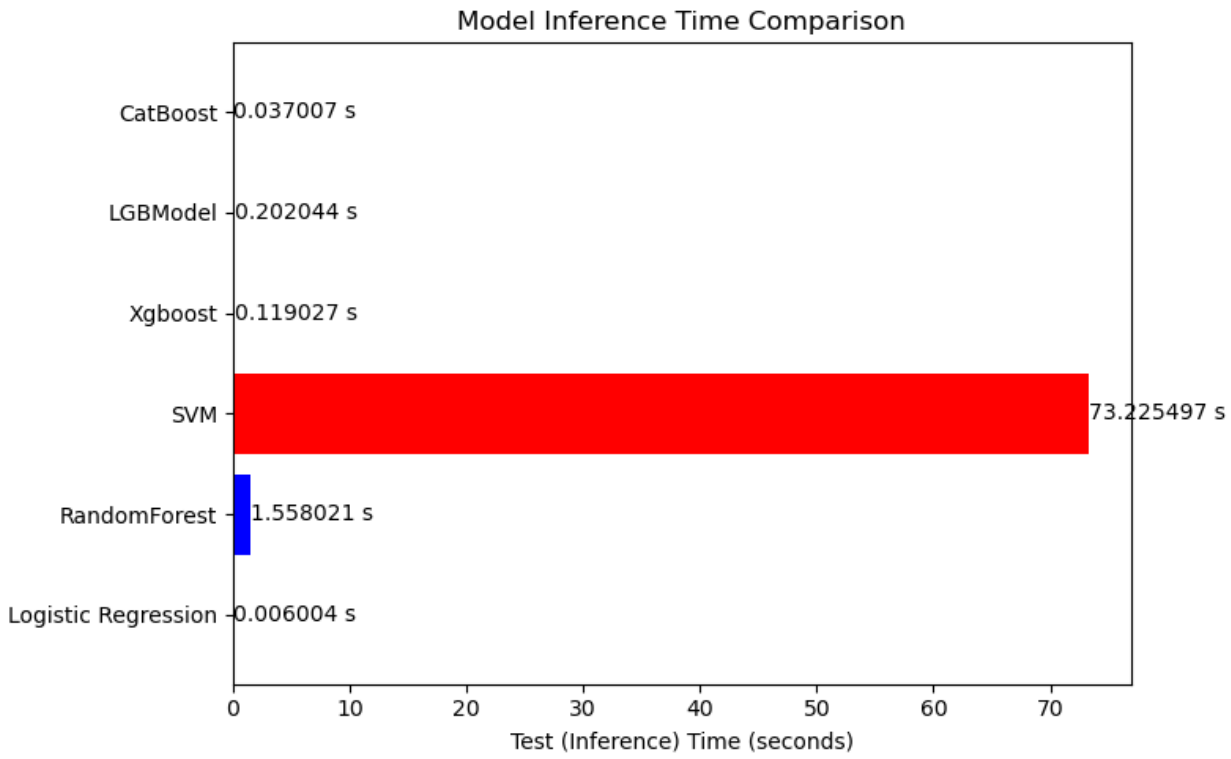
Classification Accuracy





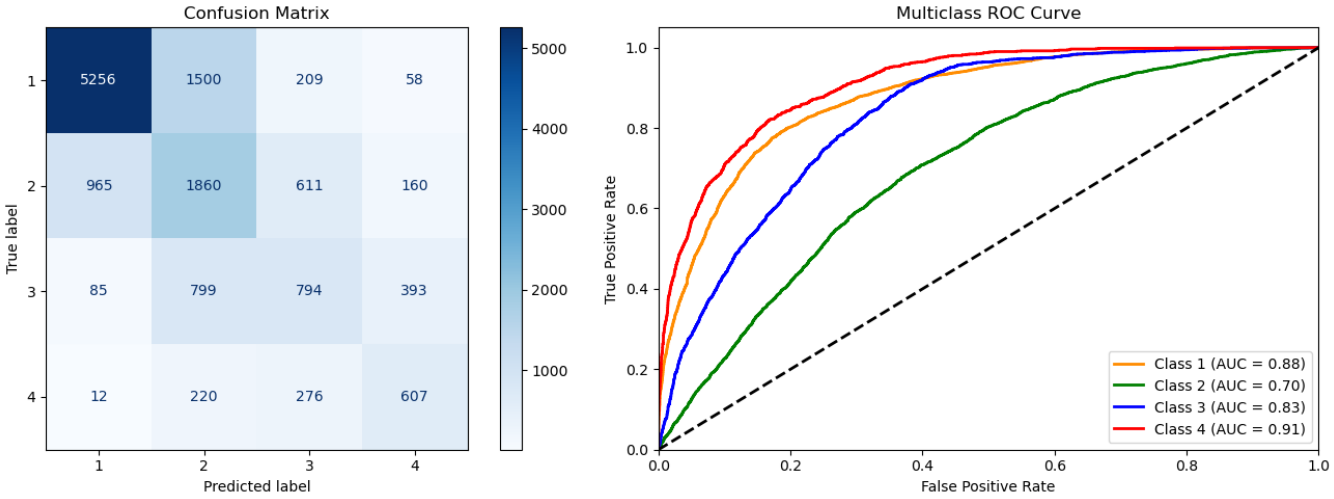
Classification Time



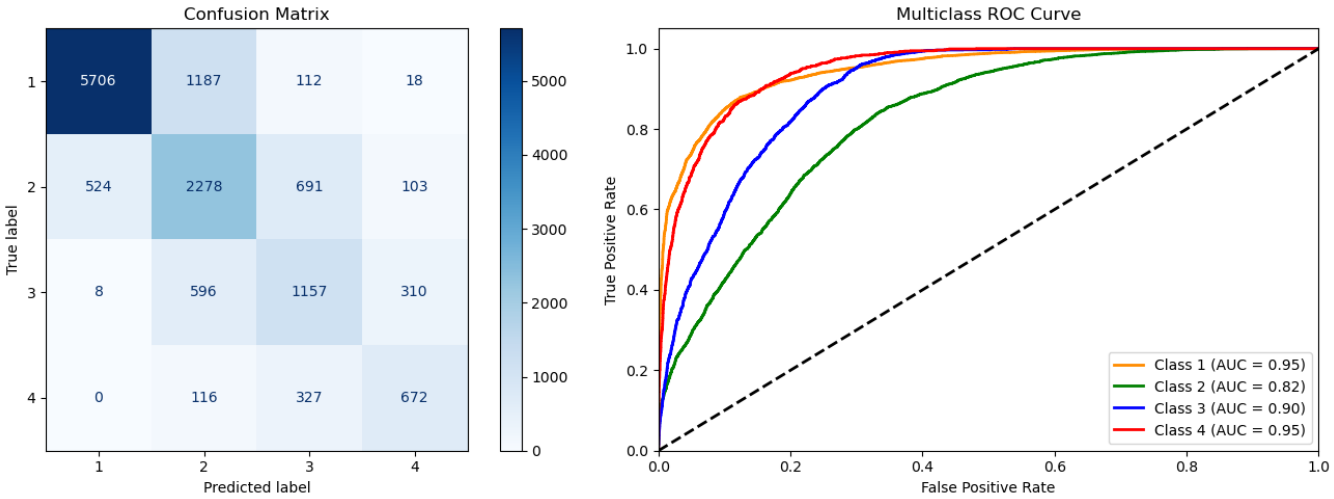


Confusion Matrix & ROC

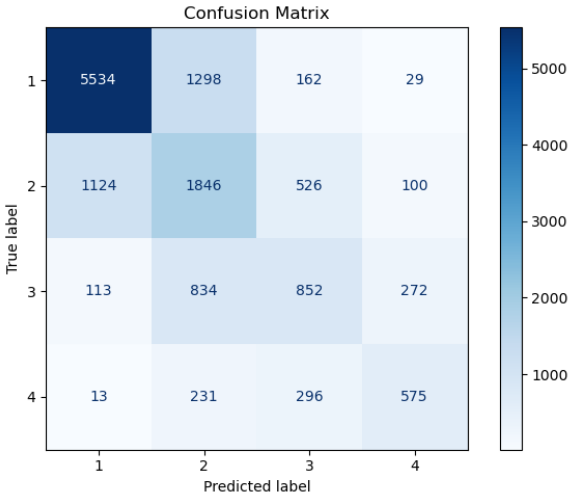
Logistic:

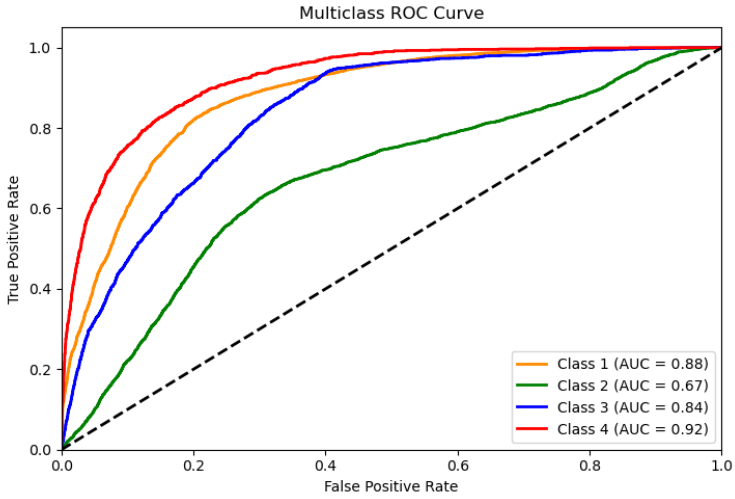


Random Forest:

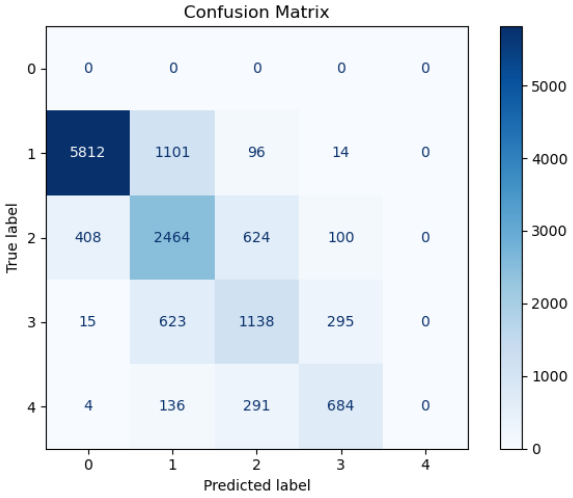


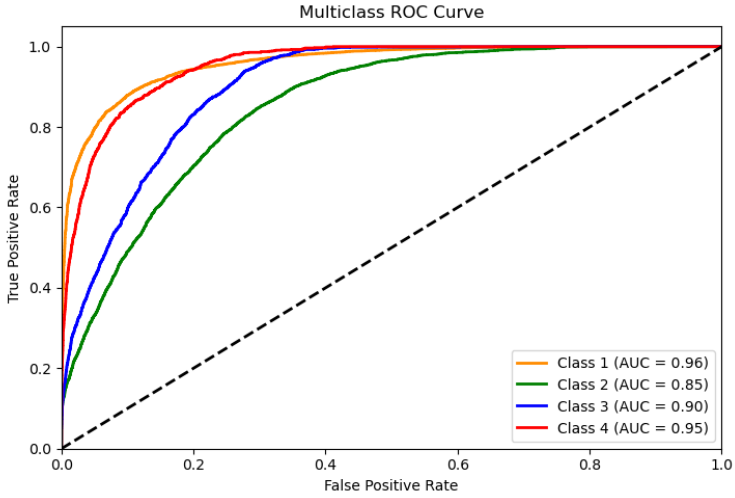
SVM:


Confusion Matrix
5534
1298
162
29
1124
1846
526
100
113
834
852
272
13
231
296
575
True label
Predicted label
1
2
3
4
5000
4000
3000
2000
1000

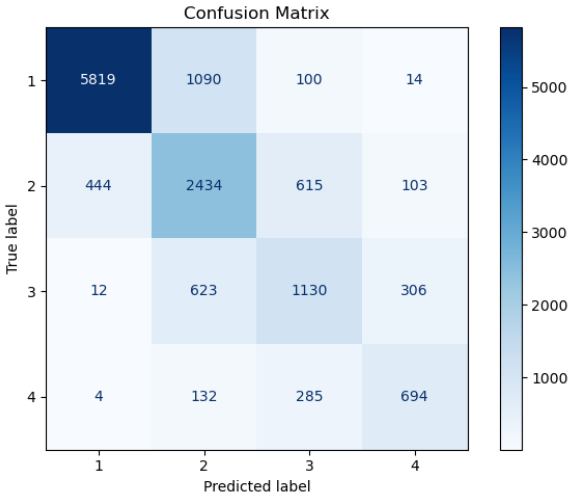

Multiclass ROC Curve
True Positive Rate
False Positive Rate
Class 1 (AUC = 0.88)
Class 2 (AUC = 0.67)
Class 3 (AUC = 0.84)
Class 4 (AUC = 0.92)
0.0
0.2
0.4
0.6
0.8
1.0

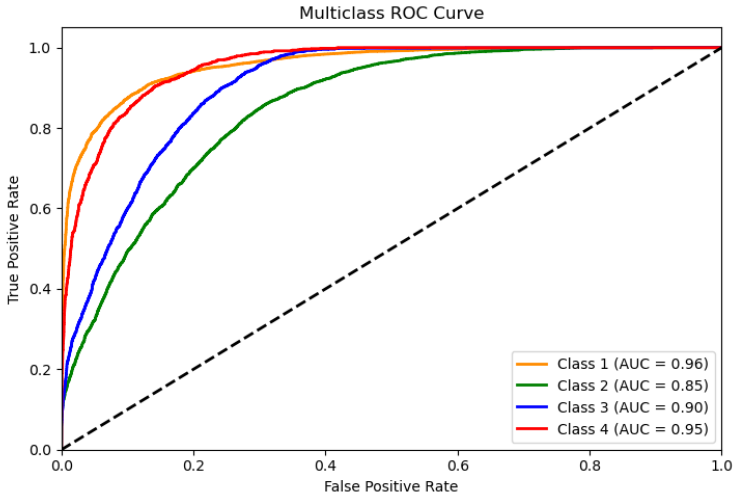
XGBoost:


Confusion Matrix
0
0
0
0
0
5812
1101
96
14
0
408
2464
624
100
0
15
623
1138
295
0
4
136
291
684
0
True label
Predicted label
0
1
2
3
4
5000
4000
3000
2000
1000
0

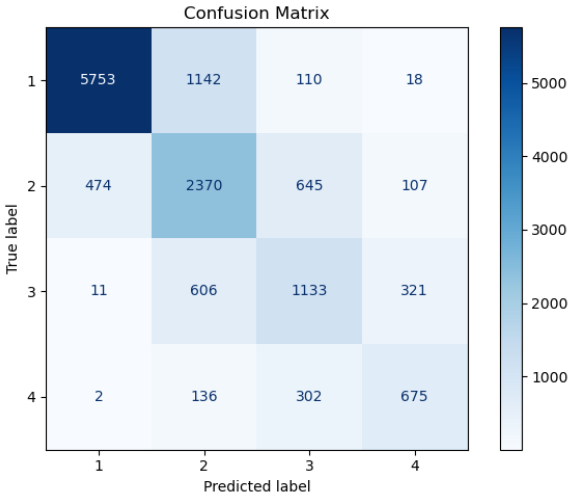

Multiclass ROC Curve
True Positive Rate
False Positive Rate
Class 1 (AUC = 0.96)
Class 2 (AUC = 0.85)
Class 3 (AUC = 0.90)
Class 4 (AUC = 0.95)
0.0
0.2
0.4
0.6
0.8
1.0

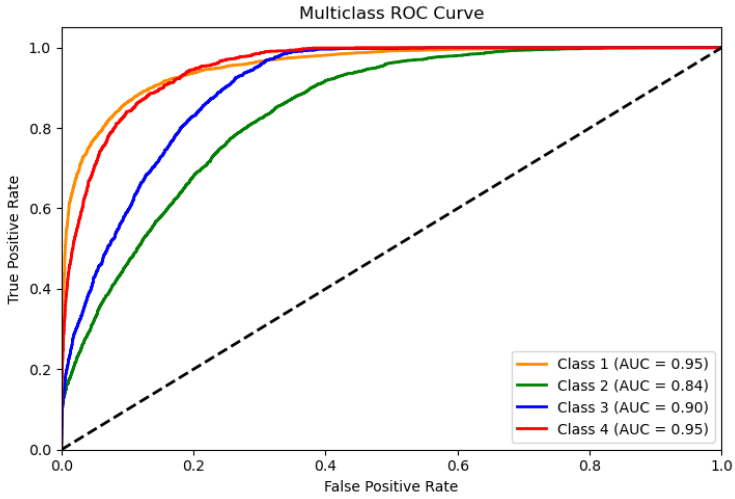
LGBM:





CatBoost:





Test Script

- We took a subset of the data (**3404 rows**) to test the Test Script.
- We trained the best model with the rest (**65628 rows**).

```
model=joblib.load(r'C:\Users\moham\Downloads\College\Machine_Learning

y_pred = model.predict(df)
acc=accuracy_score(test, y_pred)

print(acc)
print(classification_report(test, y_pred))
```

✓ 0.0s

```
0.6554054054054054
              precision    recall  f1-score   support

           1       0.87      0.82      0.85       924
           2       0.58      0.48      0.53       906
           3       0.45      0.67      0.54       732
           4       0.77      0.65      0.71       842

    accuracy                           0.66      3404
   macro avg       0.67      0.66      0.65      3404
weighted avg       0.68      0.66      0.66      3404
```

```
df.shape
```

[111] ✓ 0.0s

```
(69428, 23)
```

```
out_df=df.loc[66024:,:]
df=df.loc[:66024,:]
```

[112] ✓ 0.0s

```
(3404, 23)
```

```
df.shape
```

[187] ✓ 0.0s

```
(65628, 17)
```

```
out_df.to_csv('output.csv')
```

[113] ✓ 0.0s

- Then we did the same thing in Regression (**MS1**):

```
model = CatBoostRegressor()
model.load_model(r'C:\Users\moham\Downloads\College\Machine_Learning\P

y_pred = model.predict(df)
r2=r2_score(test, y_pred)

print("R2 Score: ",r2)
print("MSE: ",mean_squared_error(test, y_pred))
print("MAE",mean_absolute_error(test, y_pred))
```

[301] ✓ 0.0s

```
R2 Score:  0.9203678789566684
MSE:  4237381108466.8027
MAE 329092.7251957216
```

```
df.shape
```

[314] ✓ 0.0s

```
(69428, 23)
```

```
put_df=df.loc[66428:,:]
df=df.loc[:66428,:]

df.shape
```

[315] ✓ 0.0s

```
(66429, 23)
```

```
put_df.shape
```

[316] ✓ 0.0s

```
(3000, 23)
```

```
put_df.to_csv("put_df.csv")
```

[317] ✓ 0.0s

Concluding Remarks

After comparison of models in terms of (Accuracy, train/test time): **LGBM** is the best model for deployment.

Intuition (just like Phase 1):

- Features such as (name, genres, release date, price, platforms, publisher and review score) are expected to have the highest effect on our prediction.
- Features such as (achievements, trading cards and workshop support) are expected to have no significant effect.

Actual (just like Phase 1):

- The first intuition was correct.
- However, our second intuition was wrong, those features did have an effect in feature engineering better features for our prediction.