NFT COMPETITION

Predicting Old and New Total Sales Stats

Members : Muhammad Zaka Tawakal & Fitriah Hardianti

Mentor : Dr. rer. nat. Akmal Junaidi, M.Sc.

Problems

Looking for a solution to predict the total sales of NFT's using KNN Algorithm

Data

The dataset we are using is Collections.csv. The dataset contains the specific NFT item.
There are 31 variables in the dataset, namely

- primary_asset_contracts_address
- primary_asset_contracts_asset_contract_type
- primary_asset_contracts_created_date
- primary_asset_contracts_name
- primary_asset_contracts_nft_version
- primary_asset_contracts_owner
- primary_asset_contracts_schema_name
- primary_asset_contracts_symbol
- primary_asset_contracts_total_supply
- primary_asset_contracts_description
- primary_asset_contracts_dev_seller_fee_basis_points
- primary_asset_contracts_seller_fee_basis_points
- primary_asset_contracts_payout_address
- stats_one_day_volume
- stats_one_day_change
- stats_one_day_sales
- stats_one_day_average_price
- stats_seven_day_volume
- stats_seven_day_change
- stats_seven_day_sales
- stats_seven_day_average_price
- stats_thirty_day_volume
- stats_thirty_day_change

- stats_thirty_day_sales
- stats_thirty_day_average_price
- stats_total_volume
- stats_total_sales
- stats_total_supply
- stats_count
- stats_num_owners
- stats_average_price
- stats_market_cap
- stats_floor_price
- slug, stats_time
- dreated_date
- description
- display_data_card_display_style
- safelist_request_status
- name
- telegram_url
- twitter_username
- instagram_username
- discord_url
- medium_username
- external_url

First, we select and delete the variables that are not very important and leave only the variables, primary_asset_contracts_dev_seller_fee_basis_points, primary_asset_contracts_seller_fee_basis_points, stats_total_sales, and stats_total_supply.

Then, we clear columns of NaN in the primary_asset_contracts_dev_seller_fee_basis_points and

primary_asset_contracts_seller_fee_basis_points, we choose one attribute to be the dependent variable, namely stats_total_sales and we predict with KNN Algorithm.

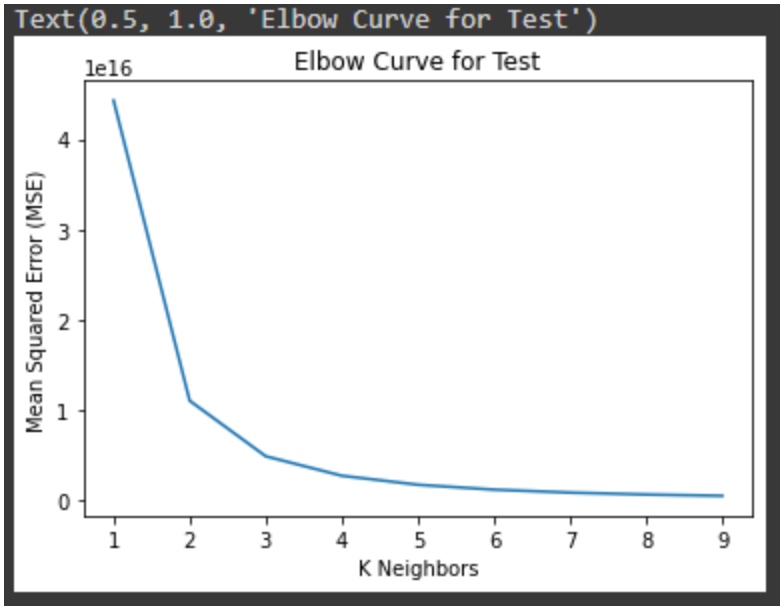
Model Build

After deleting unnecessary variables and selecting 4 variables to be predicted with KNN Algorithm. Then, deleting unneeded data/value like NaN and After that we determine the dependent variables.
The independent variable is denoted by the symbol x where the attributes that are used as independent variables are :

1. primary_asset_contracts_dev_seller_fee_basis_points
2. primary_asset_contracts_seller_fee_basis_points
3. stats_total_supply

Then, we determine the dependent variable which is symbolized by y, where the attribute used is "stats_total_sales". After splitting the data into 2 (x and y), we made Sharing data for training and validation and also built a model with the K-Nearest Neighbors algorithm. After that, determine the best K Value train model at each K value.
And then improvise by applying the value of k that produces a minimum MSE. Then, determine prediction results.

Result



Based on the graph, we see that K-Neighbors is equal to 1 while MSE is equal to 4 and starts to stabilize at K-Neighbors equal to 3.

Accuracy value of new and old models and also improvements in percentage:

```
[15] # Improvise by applying the value of k that produces a minimum MSE
     new_model  = KNN_Reg(n_neighbors = 3)

     # Train model
     new_model.fit(train_x, train_y)
     acc2 = new_model.score(test_x, test_y)

     # Prediction test
     print(' Accuracy of new model (%):', acc2*100, '\n',
           'Accuracy of old model (%):', acc1*100, '\n Improvement (%):', (acc2-acc1)*100)
```

```
 Accuracy of new model (%): -10831078999.075422
 Accuracy of old model (%): -97444450964.79669
 Improvement (%): 86613371965.72127
```

Result new and old prediction stat total sales:

```
6] # Determine the prediction results
   # Collections = ['primary_asset_contracts_dev_seller_fee_basis_points',
   #                'primary_asset_contracts_seller_fee_basis_points',
   #                'stats_total_sales', 'stats_total_supply']
   stat_sales = np.array([[750,500,250]])
   pred_old = model.predict(stat_sales)
   pred_new = new_model.predict(stat_sales)

   print(' Result Prediction Stat Total Sales Old Model: ', pred_old, '\n',
         'Result Prediction Stat Total Sales New Model: ', pred_new, )
```

```
 Result Prediction Stat Total Sales Old Model:  [49.]
 Result Prediction Stat Total Sales New Model:  [271.66666667]
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but KNeighborsRegressor was fitted with feature names
  "X does not have valid feature names, but"
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but KNeighborsRegressor was fitted with feature names
  "X does not have valid feature names, but"
```

Conclusion

We can conclude the project we made by using KNN Clustering and the variables (sprimary_asset_contracts_dev_seller_fee_basis_points, primary_asset_contracts_seller_fee_basis_points, stats_total_sales, stats_total_supply), we can get the desired outcome by using variables