Jordan food price Documentation

Model Training Assignment 2

Sief Addeen Freitekh
Dana mohammad Daoud
Abdalrhman Jad

Training multiple models to predict food price for future data to help analyse trends and patterns.

Abstract:

As discussed in Assignment 1, the data provided is about the prices of various types of food and fuel derivatives in the governorates of Jordan from 2011 to 2022.The data was divided into training data, which is all data in 2021 and before and test data, it was for the year 2022.
The document contains an explanation of the codes applied to the data to choose the best results.

Table of contents

Introduction

This documentation focuses on analysing the prices of food and fuel derivatives in the governorates of Jordan from 2011 to 2022. The aim is to develop accurate models to predict future food prices and identify trends. The documentation covers basic steps such as data coding, model selection, feature selection, and evaluation metrics. Various encryption techniques were explored and the most appropriate method determined. Regression models, including XGBoost Regression, are evaluated for accuracy. Feature selection techniques are applied to identify significant predictors. Evaluation scales designed for continuous data sets are used to evaluate model performance.

Encoding

After processing and cleaning the data in assignment 1, the data was then encoded, trying several types such as Binary, Hash, One-Hot, Target, LOOE.
This measured the accuracy percentage by using Linear Regression for each type and the most appropriate type in terms of accuracy and the appropriate number of columns.
The best result appeared to be Target Encoding.

```
encoders = {'TE': ce.TargetEncoder(),
            'OHE': ce.OneHotEncoder(),
            'BE': ce.BinaryEncoder(),
            'HE': ce.HashingEncoder(),
            'LOOE': ce.LeaveOneOutEncoder()
}
X_train_enc_results = {}
X_test_enc_results = {}
accuracy_enc = {}
for name, enc in encoders.items():
    X_train_enc_results[name], X_test_enc_results[name] = encode_X_split(enc, X_train, X_test, y_train)
    accuracy_enc[name] = lm.LinearRegression().fit(X_train_enc_results.get(name), y_train).score(X_test_enc_results.get(name), y_test)
accuracy_enc
```

✓ 20.7s

```
{'TE': 0.9840794356442698,
 'OHE': 0.9839690534195312,
 'BE': 0.3057220460844994,
 'HE': 0.06961265118006787,
 'LOOE': 0.9840660842231903}
```

ML models

Because the data here is supervised and continuous data, this is restricted to certain machine learning models, which are regression models.
Regression models will be used :

- Linear regression
- Ridge regression
- Decision trees
- Random forest
- K-nearest neighbour (KNN)
- Neural network regression
- XGBoost regression

```
df_results = pd.DataFrame(df[df['year'] == 2022]['price'])

for name, pred in predictions.items():
    df_results[name] = pred
    df_results[name] =  df_results[name].round(2)
df_results
accuracy
```

```
{'Linear': 0.8277282086479066,
 'Ridge': 0.8284145504461222,
 'Random Forest': 0.818805765271105,
 'Decision Tree': 0.8091969800960879,
 'XGBoost': 0.8318462594371997,
 'Neural Network': 0.7542896362388469,
 'KNeighbors': 0.7501715854495539}
```

When the accuracy of the above models was calculated, it was concluded that XGBoost regression is the best fit model in this dataset.

Feature Selection

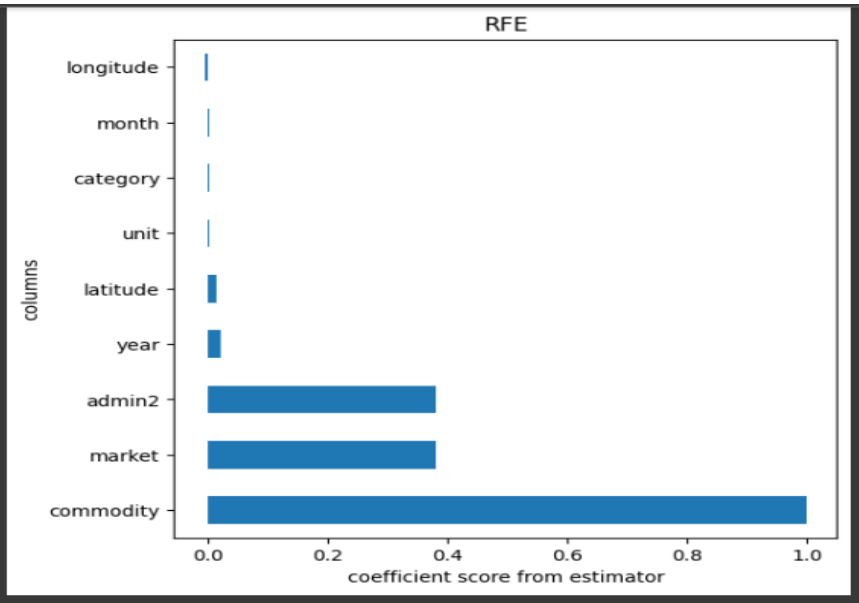
Feature selection is the process by which a subset of relevant features, or variables, are selected from a larger data set for constructing models.
Recursive Feature Elimination (RFE) and (Selectkbest) will be used in this assignment.

RFE Code:

```
#RFE (Recursive Feature Elimination)

rfe_model = fs.RFE(lm.LinearRegression(), n_features_to_select = 9)
rfe_model = rfe_model.fit(X_train_enc,y_train)
rfe_coef_score = rfe_model.estimator_.coef_
rfe_feat_cols = X_train_enc.columns[rfe_model.support_]
plt.figure(figsize = (6,6))
feat_importances_RFE = pd.Series(rfe_coef_score, index = rfe_feat_cols)
feat_importances_RFE.nlargest(9).plot(kind='barh')
plt.xlabel("coefficient score from estimator")
plt.ylabel("columns")
plt.title("RFE")
plt.show()
df_rfe = pd.DataFrame({'RFE Coefficient' : rfe_coef_score, 'RFE Selected Features' : rfe_feat_cols})
df_rfe.sort_values(by='RFE Coefficient', ascending=False)
```

RFE results:



	RFE Coefficient	RFE Selected Features
7	0.999584	commodity
3	0.379554	market
2	0.379554	admin2
0	0.019949	year
4	0.014477	latitude
8	0.001145	unit
6	0.000690	category
1	0.000621	month
5	-0.006758	longitude

As shown above, the most important features were:

1-Commodity

2-market

3-Admin2

Selectkbest Code :

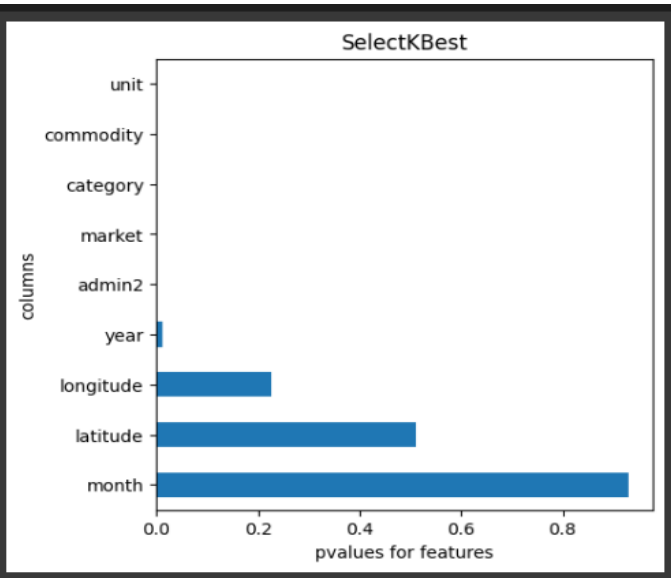
```
#SelectKBest
skb_model = fs.SelectKBest(score_func=fs.f_regression, k = 9)
skb_model.fit_transform(X_train_enc, y_train)

skb_pvalue_score = skb_model.pvalues_
skb_feat_cols = X_train_enc.columns

plt.figure(figsize = (5,5))
feat_importances_skb = pd.Series(skb_pvalue_score, index=X_train_enc.columns)
feat_importances_skb.nlargest(9).plot(kind='barh')
plt.xlabel("pvalues for features")
plt.ylabel("columns")
plt.title("SelectKBest")
plt.show()

df_skbest = pd.DataFrame({'SKBest pvalues' : skb_pvalue_score, 'SKBest Selected Features' : skb_feat_cols})
df_skbest.sort_values(by='SKBest pvalues', ascending=False)
```

Selectkbest Results :



	SKBest pvalues	SKBest Selected Features
1	9.307870e-01	month
4	5.108059e-01	latitude
5	2.255383e-01	longitude
0	1.232636e-02	year
2	8.326505e-07	admin2
3	8.326505e-07	market
6	0.000000e+00	category
7	0.000000e+00	commodity
8	0.000000e+00	unit

Here, the results showed different features from the RFE, and the order of importance was, respectively:
1- month
2- latitude
3- longitude

Evaluation Metrics

Evaluation metrics are used to measure the quality of machine learning models. Evaluating machine learning models or algorithms is essential for any project.The provided dataset contains continuous values for the target variable. Therefore, evaluation metrics suitable for this situation are Root Mean Squared Error, Mean Absolute Error, R2 Score, and Explained Variance Score.All these evaluation metrics were applied to the XGBoost model, which is the highest-accuracy model.

```
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from sklearn.metrics import explained_variance_score

xgb_model = xgb.XGBRegressor()
xgb_model.fit(X_train_enc, y_train)
y_pred = xgb_model.predict(X_test_enc)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error (RMSE):", rmse)
mae = mean_absolute_error(y_test, y_pred)
print("mean absolute error (MAE):", mae)
r2 = r2_score(y_test, y_pred)
print("r2 score:", r2)
evs = explained_variance_score(y_test, y_pred)
print("explained variance score:", evs)
```

```
Root Mean Squared Error (RMSE): 0.2592758268146776
mean absolute error (MAE): 0.14851432502597017
r2 score: 0.9887331018948239
explained variance score: 0.9894827504625239
```

Deployment

After choosing the best model, the model must be saved, then tests should be conducted on the test data, so that prices are expected in the event that new data is used, and then linking the files together to create a user interface (web application) that makes it easier for the user to understand the inputs and outputs.

Note:A well-executed deployment process ensures that ML models are effective, reliable, and usable in real-world

Python file:

```python
@app.route('/')
def home():
    return render_template('deployment.html')

@app.route('/',  methods=['POST'])
def uploadFile():
    if request.method == 'POST':
        f = request.files.get('file')
        df = pd.read_csv(f)
        X_test = df
        y_test = X_test['price']
        X_test = X_test.drop('price', axis = 1)
        xgb_model = load_pkl('models/XGBRegressor.pkl', app)
        score = xgb_model.score(X_test,y_test)
        score = round(score, 2)
        y_pred = xgb_model.predict(X_test)
        accuracy = compute_accuracy(y_test, y_pred)
        accuracy = round(accuracy, 2)
        df_y_results = pd.DataFrame({'y_true': y_test, 'y_pred': y_pred.round(2)})

        plt.figure(figsize =(10, 8))
        plt.title("Plot")
        plt.xlabel('count of data')
        plt.ylabel('price')
        plt.plot(y_pred[:100], color = 'red',linewidth=0.75)
        plt.plot(y_test[:100], color = 'blue',linewidth=0.75)
        path = os.path.join(app.root_path, 'static/new_plot.png')
        plt.savefig(path)

    return render_template("deployment.html", score = score, accuracy = accuracy, tables=[df_y_results.to_html(classes='data')],
                           titles=df_y_results.columns.values, img_url = '/static/new_plot.png')
```

Html file:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Assignment 2</title>
    <link rel = "stylesheet" href = "/static/style.css">
  </head>

  <body>
    <div class="header">
      <p class = "logo"><img src = "/static/ORANGE_LOGO_rgb-black.jpg"></p>
      </div>

      <div class = "csv_input">
      <form method = "POST" enctype="multipart/form-data" action = "/">
          <input type="file" name="file" accept=".csv">
          <input type = "submit" value="Upload">
      </form>

      <div class = "results">
        <p> Model score: {{score}}</p>
        <p> Accuracy score: {{accuracy}}</p>
      </div>

      </div>

      <br>
      <div class = "plot">
        <img src={{img_url}}>
      </div>
      <div class = "data_div">
      <div class = "output">
        {% for table in tables %}
            {{ table|safe }}
        {% endfor %}
```

Conclusion

In conclusion, this article presented an analysis of food and fuel derivative prices in the governorates of Jordan from 2011 to 2022. The objective was to develop models that could predict future food prices and identify patterns and trends. Overall, the analysis presented in this article provides valuable insights into predicting food prices in Jordan's governorates. The findings highlight the importance of proper encoding techniques, model selection, and feature selection to achieve accurate predictions. These results can be beneficial for policymakers, researchers, and stakeholders in understanding price trends and making informed decisions related to food and fuel derivatives in Jordan.