Beltone

Data and AI Hackathon Hand Out Draft

October 2024

This document is designed to introduce you and your team to the problem statement for the AI Hackathon. Please share this handout with your teammates to ensure everyone is prepared and ready to excel.

1 Introduction

In this challenge, you'll get to dive into the world of gold prices in Egypt—where economics, markets, and global events all play a role in shaping the value of this precious metal.

Predicting gold prices is no small feat; it's a puzzle that investors, traders, policymakers, and consumers are always trying to solve. The data provided for this hackathon is pulled from a wide array of sources: historical gold prices, stock market trends, economic indicators, and news articles. Each of these pieces holds clues to the bigger picture, offering insights into the dynamic forces that influence gold prices in Egypt.

Your task? To crack the code and uncover the hidden patterns within these diverse datasets. Not only will this be a great opportunity to flex your data science skills, but it also has the potential to unlock real-world economic insights. So, dive in, explore the predictive power of the data, and see how far your analysis can take you.

2 Problem Statement

Your challenge is to predict the percent change of gold price in the next day using the dataset provided. The dataset comprises of multivariate numerical data, textual news data, and time-series data. Detailed descriptions of the dataset can be found in the following section.

3 Dataset Description

This dataset provides a comprehensive overview of various factors related to Egypt's economy and financial landscape. It consists of eight tables. Each data point within these tables is timestamped, ensuring alignment across the different data sources. Your target variable is in target_gold.csv. You are not allowed to extract any features from this table, it will not be included in the test data.

3.1 TARGET_GOLD.CSV

This is your target variable, with the percentage change representing your target column. This data set contains the daily percentage change of price in gold. For example, if the price is 100 on the 1st of October and 102 on the 2nd of October, the data will show 2% on the 1st of October. You are not allowed to extract any features from this table, it will not be included in the test data.

Column	Description	Data Type
Date		date
pct_change	Percent change of price in the next day	numeric

3.2 INTRADAY_GOLD.CSV

This contains samples of prices of gold in Egypt in different timestamps throughout the day. You can use this csv to get the closing price of each day as well as the OHLC data if available.

Column	Description	Data Type
Timestamp		timestamp
24K	Price of 24K gold in Egypt	numeric

3.3 STOCKS_PRICES_AND_VOLUMES.CSV

This dataset contains the closing price and the volume traded daily of 15 important stocks in the Egyptian stock exchange. You can use this dataset as individual stocks or aggregate the stocks to get the performance of separate industries or Egyptian stock exchange in general.

The closing price is not adjusted, meaning that all stock adjustment and corporate actions are not considered.

Columns	Description	Type
Date		date
stock_<n>_<industry>_close_price	Stock number <n> of industry <industry> closing price	numeric
stock_<n>_<industry>_volume	Stock number <n> of industry <industry> volume; number of shares traded in respective day	numeric

3.4 INFLATION_MONTH_ON_MONTH.CSV

This dataset contains inflationary figures issued monthly by the Central Bank of Egypt. The inflation rate is calculated on a month-on-month basis.

Columns	Description	Type
Date	month	date
Headline (m/m)	Total inflation including all items	numeric
Core (m/m)	Inflation excluding volatile items like food and energy, providing a clearer picture of long-term trends	numeric
Regulated Items (m/m)	Inflation of regulated items, such energy, fuel, baladi bread, wheat… etc	numeric
Fruits and Vegetables (m/m)	Inflation of fruits and vegetables	numeric

3.5 INFLATION_YEAR_ON_YEAR.CSV

This dataset contains inflationary figures issued monthly by the Central Bank of Egypt. The inflation rate is calculated on a year-on-year basis.

Columns	Description	Type
Date	month	date
Headline (y/y)	Total inflation including all items	numeric

Core (y/y)	Inflation excluding volatile items like food and energy, providing a clearer picture of long-term trends	numeric
Regulated Items (y/y)	Inflation of regulated items, such energy, fuel, baladi bread, wheat… etc	numeric
Fruits and Vegetables (y/y)	Inflation of fruits and vegetables	numeric

3.6 EGYPTIAN_CORRIDOR_INTEREST_RATE.CSV

This table shows the historical changes in the Egyptian corridor rates as published by the Central Bank of Egypt. It includes the date of publication, the borrowing rate, and the lending rate.

Columns	Description	Type
Date	Date of publication	date
Overnight Deposit Rate	Yearly interest rate on deposits	numeric
Overnight Lending Rate	Yearly interest rate on lending	numeric

3.7 NEWS.CSV

This dataset contains news relating to the Egyptian economy and stock market. All news articles were acquired in Arabic and subsequently translated. Please note that if an article has more than one tag, it will be duplicated with a different tag. Hence, mind the data granularities. All names and entities are not translated.

Columns	Description	Type
id	Unique identifier for each article	numeric
source	Source or agency providing the article	text
category	Category from which the news was acquired	text
date	Date and time of news article	timestamp
translated_title	Translated Title from Arabic to English	text
tag	Internal tagging of news article into one or more of the following: Egyptian_Economy, Egyptian_stock_exchange, Geopolitical_tensions, Gold, and Inflation_in_egypt	text
summary	Summary of news article as generated by LLMs	text

3.8 VIX.CSV AND VXEEM.CSV

This table includes historical volatility figures for the US Stock Market volatility index (VIX) as well as the Emerging Markets Volatility index (VXEEM) collected by the Chicago Board Options Exchange (CBOE). It is a real-time market index representing the market's expectations for volatility over the coming 30 days. It is often referred to as the "fear gauge" or "fear index" because it reflects the level of uncertainty or risk perceived by investors in the stock market.

Columns	Description	Type

Date	Date of publication	date
OPEN	Index calculated on open prices	numeric
HIGH	Index calculated on high prices	numeric
LOW	Index calculated on low prices	numeric
CLOSE	Index calculated on close prices	numeric

3.9 EFFECTIVE_FEDERAL_FUNDS_RATE.CSV

This table includes historical Effective Federal Funds Rate (EFFR) in the United States.

Columns	Description	Type
Date		date
EFFR	Effective federal funds rate	numeric

3.10 HOUSING_INDEX.CSV

CSUSHPINSA refers to the U.S. National Home Price Index (NSA) published by the Federal Reserve Bank of St. Louis. It stands for Case-Shiller U.S. National Home Price Index, Not Seasonally Adjusted.

This index tracks the value of single-family home prices across the United States, offering insights into the national real estate market and long-term trends in home prices. It is a popular measure used to assess real estate market health and is often cited in discussions about housing affordability, market conditions, and price appreciation.

Columns	Description	Type
Date	month	date
CSUSHPINSA	U.S. National Home Price Index (NSA)	numeric

3.11 CRUDE_OIL_PRICES.CSV

This table includes historical crude oil price for the West Texas Intermediate (WTI) oil price as well as the Europe Brent crude oil price.

Columns	Description	Type
Date		date
WTI Oil Price FOB (Dollars per Barrel)	West Texas Intermediate oil prices	numeric
Europe Brent Crude Oil (Dollars per Barrel)	Europe Brente oil prices	numeric

4 SUCCESS CRITERIA

Each team will be judged according to specific numerical metrics as well as qualitative metrics. Your model will be evaluated using a hidden testing set that is not included within the training data you are provided. You're encouraged to split the data that you are provided into a training subset and a validation subset and then, train your models on all the data before submitting accordingly.

4.1 Quantitative Metrics

- Root Mean Squared Error
- Mean Directional Accuracy
- Bucketized F1 Score with the following buckets:
    - x < -1.5
    - -1.5 < x < -0.5
    - -0.5 < x < 0.5
    - 0.5 < x < 1.5
    - x > 1.5
- Inference Time

5 Submission Guidelines

5.1 Submission Portal

You can submit a zip file of your solution on https://beltone-ai-hackathon.com/ using the provided username and password for your team.

Each submission is automatically run, and the scores calculated, and a near-live dashboard is available for the scores for all teams.

Each submission's maximum runtime is 10 minutes per submission. If the submission is taking more than 10 minutes, it will be discarded and not scored. You can view your submission status and error logs from the dashboard as well.

Each team has a maximum of 3 submissions per hour. Feel free to submit as much as needed as long as you are not submitting more than the allowable hourly limit.

5.2 Submission Structure

Submission will be done through a zip file containing the following structure.

Zip file to contain		Type	Description
hackathon_run.py		py	Main py file of your submission that will be called in order to run your code on our test set
Pickles		folder	Folder with all pickle files required to run your code such as your ML models
	pkl1	pickle	
	pkl2	pickle	
	pkl…	pickle	
	pkln	pickle	
InputData		folder	Folder with input data to your script

	csv1	csv	
	csv2	csv	
	csv…	csv	
	csvn	csv	
Scripts		folder	Folder with any other py scripts your hackathon_run.py file requires
	py1	py	
	py2	py	
	py…	py	
	pyn	py	
Libraries		folder	Folder with any required library outside the main provided libraries
	L1	folder	
	L2	folder	
	L…	folder	
	Ln	folder	
Workout		folder	Folder with any extra files you want to send, such EDA, notebooks, presentations… etc.

5.2.1 hackathon_run.py

This is your main code that will be run on the test set to acquire your predictions. This code should not be training any models, instead you should save your models in pickle files in Pickles folder.

Your script should do the following steps:

- Read data from the input_path argument.
- The script should not read the target_gold.csv. This information is your required to output.
- Prepare and process the data to be fed into your models.
- Read the pickle files for your models.
- Produce the output data frame.
- Save the output file to the output_path.

You can have other python scripts that are imported in hackathon_run.py if needed. These scripts should be in Scripts folder.

If you require any other libraries than the ones provided, please include them in the Libraries folder and import them from their respective path.

The test is similar to the train set you have been provided but with different dates, thus all preprocessing steps done on the train test will work on the test set as well.

5.2.2 Arguments

Your hackathon_run.py should take 2 arguments, input_path and output_path. The input_path is the path on which the input data to your models is stored. The output_path is the path on which your predictions will be stored. You can use the following code example to read arguments.

```
import argparse

import os

import pandas as pd

# Create an argument parser object

parser = argparse.ArgumentParser(description="Process input and output file paths.")

# Define the input and output path arguments

parser.add_argument('--input_path', type=str, default='InputData/', required=True, help='Path to the input file.')

parser.add_argument('--output_path', type=str, default='predictions.csv', required=True, help='Path to the output file.')

# Parse the arguments

args = parser.parse_args()

# Access the input and output paths

input_path = args.input_path

output_path = args.output_path

# Print them (for verification)

print(f"Input Path: {input_path}")

print(f"Output Path: {output_path}")

# Read news data from input

df = pd.read_csv(os.path.join(input_path,'news.csv'))

# Write data to output

df.to_csv(output_path), index = False)
```

5.2.3 Pickle Files

You are required to train your models on your machine and save it in pickle files. Your main script should be calling on these pickle files and running them on the data as well as providing an output csv for the prediction.

To save a variable to a pickle file:

```
import pickle

# Example variable to save

data = {'key': 'value', 'numbers': [1, 2, 3]}

# Open a file in write-binary mode ('wb')
```

```
with open('data.pickle', 'wb') as f:

            # Use pickle.dump() to serialize and save the variable

            pickle.dump(data, f)
```

To read a pickle file:

```
import pickle

# Open the file in read-binary mode ('rb')

with open('data.pickle', 'rb') as f:

   # Use pickle.load() to deserialize the variable

   loaded_data = pickle.load(f)

print(loaded_data)
```

5.2.4 Python Environment

Your code will run on a python3 environment with the following libraries installed. If you want to use any other library, you have to include it in the Libraries folder.

```
psycopg2-binary==2.9.9
SQLAlchemy==2.0.35
et-xmlfile==1.1.0
lxml==5.3.0
openpyxl==3.1.5
pyarrow==17.0.0
autopep8==2.3.1
pymssql==2.3.1
xlsxwriter==3.2.0
imbalanced-learn==0.12.3
Levenshtein==0.26.0
boto3==1.35.10
pandas==2.2.3
numpy==1.26.4
scikit-learn==1.5.2
xgboost==2.1.1
lightgbm==4.5.0
statsmodels==0.14.3
pmdarima==2.0.4
prophet==1.1.5
pystan==3.10.0
tsfresh==0.20.3
nltk==3.9.1
spacy==3.7.6
transformers==4.44.2
requests==2.32.3
matplotlib==3.9.2
seaborn==0.13.2
torch==2.4.1
tensorflow==2.17.0
keras==3.5.0
```

You can install all these libraries by following these steps:

1. Create a text file called requirements.txt and copy the libraries and versions above in this text file.

2. Run the following code snippet:

```
import subprocess

import sys

def install_requirements(requirements_file='requirements.txt'):

    try:

        # Use subprocess to run the pip install command

        subprocess.check_call([sys.executable, '-m', 'pip', 'install', '-r', requirements_file])

        print(f"Successfully installed packages from {requirements_file}")

    except subprocess.CalledProcessError as e:

        print(f"Failed to install packages from {requirements_file}. Error: {e}")

# Call the function with the path to your requirements.txt

install_requirements('requirements.txt')
```

5.2.5 Importing other libraries

If you need to import any other libraries apart from the ones mentioned above, add the library folders to Libraries folder and import it from you hackathon_run.py normally as you would import any other python scripts.

5.2.6 Output File

Your script should output 1 file to the output_path called predictions.csv with only 2 columns: 'date' and 'prediction'. Please make sure that you specify index = False.

5.2.7 Testing your code

We highly recommend testing your code using shell scripts before submitting to make sure it reads the data from input_path and write data to output_path. You can run the following command to test your code arguments from Jupyter Notebook:

```
!python hackathon_run.py --input_path /path/to/input/files --output_path /path/to/output/file.csv
```

5.2.8 General Guidelines

- Your maximum runtime 10 minutes on ml.m5.xlarge aws instance.
- If any of your preprocessing steps or models require looking back at previous dates, do not use a look back period of more than 6 weeks.
- You cannot use same day features to predict close price; for example, you cannot use stock hist data of 12th January to predict gold prices of 12th January.
- The target variable is already shifted to match the previous note.
- If a submission has a wrong shift strategy, score will be 0, even if it passed the initial scoring calculation that can be viewed from the dashboard.
- You do not have to use all the provided datasets, feel free to use any combination of them, however, you are not allowed to use any other external data sets.
- Submission zip file cannot exceed 100 MB, you do not have to submit any of your training data in order to save on size

6 LEADERBOARD AND SCOREBOARD DASHBOARD

To track your team's progress during the Hackathon, we have created a live performance leaderboard. You can access the leaderboard using your credentials, which will be sent to you via email. This dashboard will allow you to view the rankings of all teams, track your top submissions, and monitor key metrics such as execution time, final scores, and other specific evaluation metrics. Additionally, the dashboard provides a detailed breakdown of your submission history, including the status of each run, evaluation feedback, and a visual representation of your score trends over time. This tool is designed to help you refine your models effectively and stay informed about your standing in the competition. The link to the dashboard will be sent to you via emails when it is live.