Tarefa#3 - Aprendizado supervisionado - MO432

June 27, 2021

Universidade Estadual de Campinas (UNICAMP), Instituto de Computação (IC)

Prof. Jacques Wainer, 2021s1

[ ]:
```
# RA & Name
print('264965: ' + 'Décio Luiz Gazzoni Filho')
print('265673: ' + 'Gabriel Luciano Gomes')
print('192880: ' + 'Lucas Borges Rondon')
```

```
264965: Décio Luiz Gazzoni Filho
265673: Gabriel Luciano Gomes
192880: Lucas Borges Rondon
```

1 Metodologia

Para este projeto, foi utilizado o scikit-learn e Statsmodels para explorar os modelos de regressores e classificadores e o Keras (TensorFlow) para o modelo de rede neural recorrente. As seções abaixo descreverão todo o processo realizado neste trabalho.

1.1 Leitura e Pré-processamento

Os dados foram adquiridos por meio de um link divulgado pelo professor, que correspondem em um conjunto de 1096 amostras, com dois atributos (data e preço), que são o preço do ouro e a data, a cada semana, desde 18/06/2000. Como estamos interessados apenas no valor do ouro, a coluna de data foi descartada.

Foi tentado realizar um experimento de consideração de porcentagens (razões) entre os valores da semana corrente e a anterior. Entretanto, nenhum resultado favorável foi atingido. Dessa forma, os dados foram utilizados sem nenhum tipo de pré-processamento.

1.2 Janela deslizante

Como neste projeto é explorado aspectos de séries temporais, o conjunto de dados adotou uma técnica chamada "Janela Deslizante". Esta técnica é responsável por separar instâncias do conjunto para formar uma sequência que possibilita e facilita a regressão/classificação dos dados.

Todos os modelos utilizados apresentados neste trabalho exploram a ideia de janela deslizante, exceto os modelos de RNN e o ARIMA. Isto porque eles exploram conceitos diferentes para tratar essa técnica, como o lookback (nas RNNs) e memória implícita (ARIMA).

1.3 Otimização de Hiperparâmetros

Como diversos dos modelos explorados utilizam hiperparâmetros, foi necessário utilizar algum algoritmo de busca para identificar os melhores valores destes hiperparâmetros. Para isso, foi utilizado a biblioteca 'hyperopt', que consiste em uma otimização Bayesiana. Além das configurações de cada modelo, o tamanho da janela também foi explorada neste otimizador a fim de obter o melhor resultado possível dos modelos.

2 Benchmarks

Como os preços do ouro no mercado é compreendido como um *random walk*, o modelo de persistência para regressão prediz que o preço corrente é o mesmo que o da semana anterior. Esse é o modelo a ser superado.

Para a classificação de um *random walk*, com média zero, ambas os resultados (subir ou descer) são equiprováveis e, portanto, não haveria uma melhor estratégia a ser adotada. Porém, observa-se um 'vies' de subida dos preços, ao longo do tempo, provavelmente podendo ser atribuído à inflação. Neste caso, a probabilidade de subida é maior do que a de descida e a melhor previsão é que o preço sempre sobe.

2.1 Modelos de Regressão e Classificação Utilizados

Para realizar o estudo, os seguintes modelos foram explorados:

- Regressão
- SVM
- Ridge
- Lasso
- GBM
- Random Forest
- Decision Tree
- ARIMA
- LSTM
- SimpleRNN
- MLP
- Classificação
- SVM
- Ridge
- GBM
- Random Forest
- Decision Tree

- ARIMA
- MLP

Além do uso da classificação direta, foi explorado o uso de regressores como classificadores indiretos. Isto para testar a hipótese não só de que um regressor pode apresentar melhor resultado para classificação, mas também porque existem modelos que não possuem a classificação, como ARIMA e Lasso. Sendo eles: - SVM - Ridge - Lasso - GBM - Random Forest - Decision Tree - ARIMA

3 Resultados

Treino

3.0.1 Regressão

[8]:
```
import pandas as pd

pd.set_option('display.max_colwidth', 0)
df = pd.read_csv('/content/drive/Shareddrives/MO432/regressao_2.csv')
df
```

[8]:

	name	…	default
0	Previous Times Constant	…	False
1	Previous Times Constant	…	True
2	ARIMA	…	False
3	ARIMA	…	True
4	Lasso L1	…	False
5	Ridge L2	…	False
6	Lasso L1	…	True
7	Ridge L2	…	True
8	MLP	…	True
9	MLP	…	False
10	LSTM	…	False
11	RNN	…	False
12	GBM	…	False
13	GBM	…	True
14	Random Forest	…	True
15	Random Forest	…	False
16	Decision Tree	…	False
17	Decision Tree	…	True
18	SVM com RBF	…	False
19	SVM com RBF	…	True

[20 rows x 4 columns]

3.0.2 Classificação com Regressores

[10]:
```
df = pd.read_csv('/content/drive/Shareddrives/MO432/
↪classificacao_usando_regressao_2.csv')
df
```

[10]:
```
              name  … default
0   ArimaRegressor  …   False
1   MLP             …    True
2   Lasso L1        …   False
3   SVM com RBF     …   False
4   ArimaRegressor  …    True
5   Ridge L2        …   False
6   Ridge L2        …    True
7   Lasso L1        …    True
8   MLP             …   False
9   GBM             …   False
10  SVM com RBF     …    True
11  Decision Tree   …    True
12  GBM             …    True
13  Decision Tree   …   False
14  Random Forest   …   False
15  Random Forest   …    True

[16 rows x 4 columns]
```

3.0.3 Classificação Direta

[9]:
```
df = pd.read_csv('/content/drive/Shareddrives/MO432/classificacao_2.csv')
df
```

[9]:
```
              name  … default
0   SVC com RBF     …   False
1   Ridge L2        …   False
2   Ridge L2        …    True
3   Always 1        …   False
4   Always 1        …    True
5   Decision Tree   …   False
6   MLP             …   False
7   MLP             …    True
8   SVC com RBF     …    True
9   GBM             …   False
10  Random Forest   …   False
11  Random Forest   …    True
12  GBM             …    True
13  Decision Tree   …    True
```

```
[14 rows x 4 columns]
```

3.0.4 Análise dos resultados

No caso da regressão, não foi possível encontrar um resultado melhor do que aquele que repete o preço da semana anterior, com pequeno acrescimo em função do pequeno viés de subida dos dados. A classificação com regressão, também não obteve bons resultados, se comparado à classificação direta. Por sua vez, o único resultado considerável na classificação direta, foi utilizando o modelo SVC, com acurácia de **56%**, significativamente superior à previsão de subida contínua (Always 1), com **52,8%**.

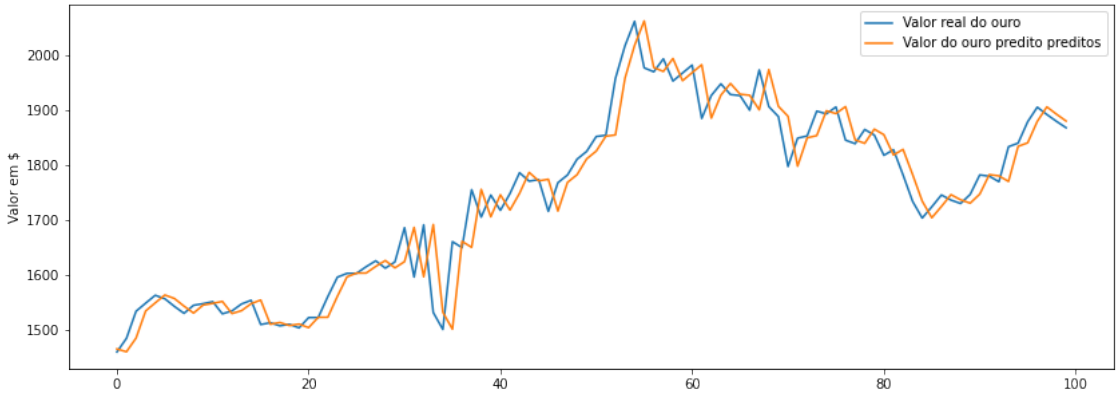
3.1 Teste

3.1.1 Regressão

[14]:
```
import matplotlib.pyplot as plt

df_regressao = pd.read_csv('/content/drive/Shareddrives/MO432/pred_regressao.
 ↪csv')

plt.figure(figsize=(14,5))
plt.plot(df_regressao['x_test'], label = 'Valor real do ouro')
plt.plot(df_regressao['y_pred'], label = 'Valor do ouro predito preditos')
plt.ylabel('Valor em $')
plt.legend()
plt.show()
```



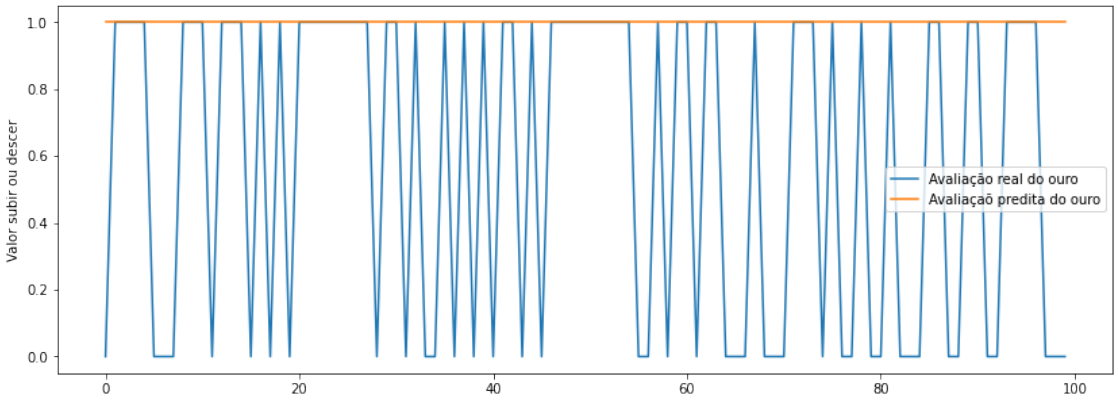
RMSE de 43.846886993100505 para o conjunto de teste.

3.1.2 Classificação

[15]:
```
import matplotlib.pyplot as plt

df_class = pd.read_csv('/content/drive/Shareddrives/MO432/pred_classificacao.
 ↪csv')

plt.figure(figsize=(14,5))
plt.plot(df_class['y_test'], label = 'Avaliação real do ouro')
plt.plot(df_class['y_pred'], label = 'Avaliaçaõ predita do ouro')
plt.ylabel('Valor subir ou descer')
plt.legend()
plt.show()
```



Acurácia de 0.58 para a base de teste

Como discutido anteriormente, o melhor regressor apenas prevê na semana corrente o mesmo preço da semana anterior, com uma pequena correção multiplicativa. Já o classificador SVC, ao analisar sua saída, constatou-se que este previu que os preços sempre subiriam. Portanto, apenas de todos os esforços de todos os diferentes regressores e classificadores, confirmou-se o esperado para um processo do tipo *random walk*, que não é possível fazer uma previsão/classificação ingênua.