Universidade Federal de Viçosa

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

Principais Decisões

[1] - Horizonte de atuação:

O primeiro passo foi definir como funcionaria o nosso preditor. Tínhamos duas possibilidades: a primeira seria permitir ao usuário fornecer uma entrada de dados manualmente e, através dessa entrada, realizar uma predição do preço de fechamento daquele dia. Já a outra hipótese se baseia em prever o preço de fechamento do dia posterior, visto que temos os dados até o dia corrente.

Introduzidas as duas abordagens, analisando o formato dos nossos dados e, somando a isso, na tentativa de implementar uma ideia mais "ousada", optamos por utilizar a segunda abordagem como horizonte para desenvolvimento deste projeto. Dessa forma, a nossa aplicação prevê com 1 dia de antecedência o preço de fechamento do *Bitcoin.*

[2] - Aquisição de dados:

A aquisição de dados da nossa aplicação é feita via uma API do *Yahoo Finance.* A biblioteca utilizada é a `pandas_datareader`.Os dados dessa API são atualizados diariamente e, dessa forma, todo dia podemos realizar uma predição do *Bitcoin* amanhã. Em relação ao período de dados, obtemos dados a partir do ano de 2018.

Date	High	Low	Open	Close	Volume	Adj Close
2018-01-01	14112.200195	13154.700195	14112.200195	13657.200195	10291200000	13657.200195
2018-01-02	15444.599609	13163.599609	13625.000000	14982.099609	16846600192	14982.099609
2018-01-03	15572.799805	14844.500000	14978.200195	15201.000000	16871900160	15201.000000
2018-01-04	15739.700195	14522.200195	15270.700195	15599.200195	21783199744	15599.200195
2018-01-05	17705.199219	15202.799805	15477.200195	17429.500000	23840899072	17429.500000
...	...	...	...	...	...	...
2022-03-23	42893.507812	41877.507812	42364.378906	42892.957031	25242943069	42892.957031
2022-03-24	44131.855469	42726.164062	42886.652344	43960.933594	31042992291	43960.933594
2022-03-25	44999.492188	43706.285156	43964.546875	44348.730469	30574413034	44348.730469
2022-03-26	44735.996094	44166.273438	44349.859375	44500.828125	16950455995	44500.828125
2022-03-27	44859.601562	44449.078125	44505.839844	44820.730469	19630307328	44820.730469

Figura 1: Formato dos dados obtidos via API.

Universidade Federal de Viçosa

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

[3] - Preparação dos dados:

Na etapa de *Data Preparation,* a nossa aplicação leva em consideração alguns aspectos:

1) **Limpeza de *outliers:*** em um processo tradicional de construção de um modelo de *machine learning,* é de extrema importância a implementação de uma etapa de limpeza dos *outliers*. Entretanto, a nossa aplicação não contém um fator externo que possa cometer erros (como um erro de medição de um sensor em um processo industrial, por exemplo). Dessa forma, não julgamos relevante a implementação de uma etapa de remoção dos *outliers.* Além disso, também assumimos que as grandes variações são interessantes para o processo de construção de informação do nosso modelo, vide a nossa aplicação no mercado acionário.

2) **Normalização dos dados:** inicialmente, tínhamos uma ideia de testar modelos de *deep learning* como *LSTM*, *GRU* e *MLP* para testes. Dessa forma, a literatura expõe que tais modelos funcionam melhor com dados normalizados e, sendo assim, foram implementadas funções normalizadoras por *mínimos e máximos* e também por *desvio padrão.*
3) **Ajuste de *target:*** observando a Figura 1 temos apenas o preço de fechamento do último dia. Como queremos prever o preço de fechamento do dia posterior(chamamos de *Close_Tomorrow*) ao último, precisamos criar essa coluna através do `pandas.shift(-1.`Há também uma nova coluna chamada *Close_Yesteday* que nos diz o preço de fechamento do dia anterior. Logo, a nossa *target* agora é a coluna ***Close_Tomorrow.***

Date	High	Low	Open	Close	Volume	Close_Yesterday	Close_Tomorrow
2018-01-01	-0.324967	-0.337359	-0.305377	13657.200195	-0.749441	NaN	14982.099609
2018-01-02	-0.253006	-0.336850	-0.332397	14982.099609	-0.432707	13657.200195	15201.000000
2018-01-03	-0.246082	-0.240665	-0.257350	15201.000000	-0.431484	14982.099609	15599.200195
2018-01-04	-0.237068	-0.259108	-0.241128	15599.200195	-0.194187	15201.000000	17429.500000
2018-01-05	-0.130915	-0.220162	-0.229676	17429.500000	-0.094766	15599.200195	17527.000000
...	...	...	...	...	...	...	...
2022-03-23	1.229461	1.306229	1.261455	42892.957031	-0.027024	42358.808594	43960.933594
2022-03-24	1.296342	1.354791	1.290419	43960.933594	0.253215	42892.957031	44348.730469
2022-03-25	1.343201	1.410876	1.350198	44348.730469	0.230575	43960.933594	44500.828125
2022-03-26	1.328970	1.437198	1.371567	44500.828125	-0.427689	44348.730469	46881.011719
2022-03-28	1.466905	1.583099	1.508944	46881.011719	0.213118	44500.828125	NaN

Figura 2: Criação das novas colunas.

Universidade Federal de Viçosa

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

[4] - *Feature engineering*:

Como evidenciado na Figura 1, nosso conjunto inicial de dados possui apenas 5 variáveis de entrada. Dessa forma, criamos algumas *features* para acrescentar informação aos modelos a serem testados. Tais *features* foram:

- **Derivada;**
- **Integral;**
- **Momentos estatísticos;**
- **Combinações polinomiais;**
- **Subtrações entre variáveis.**

Derivada

A derivada surge com intuito de adicionar informações de taxa de variação ao dataset. Para fazer isso, foi feito uso da função *diff*, que faz a diferença entre as linhas de forma par-a-par, considerando cada uma das colunas do dataset.

```
1. def derivada(self):
2.         for column in self.data.columns:
3.             if column != self.target:
4.                 self.df_fe[f'{column}_derivative'] =
                   self.data[column].diff()
```

Assim teremos uma nova coluna que tem a terminação “derivative” para cada uma das nossas colunas de base.

Integral

A integral surgiu da ideia de *data driven* que o grupo escolheu ter como abordagem, assim, foram feitas algumas operações com intuito de checar, experimentalmente, se teríamos bons resultados. A adição da ideia de integral foi uma das experimentações e sua aplicação foi na utilização de uma janela de três dias para realização da soma dos valores das colunas. Assim, supondo que uma linha seja referente ao dia 23/04, então as colunas dessa linha serão somadas as colunas correspondentes nos 3 últimos dias.

```
1. def integral(self):
2.         for column in self.data.columns:
3.             if column != self.target:
4.                 #Integral numa janela de 3 dias.
5.                 self.df_fe[f'{column}_integral'] =
                   self.data[column].rolling(3).sum()
```

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

Momentos estatísticos

Quanto a momentos estatísticos, foi escolhido pelo grupo trabalhar com a média móvel e o desvio padrão móvel. Essa escolha se deu principalmente pela natureza variável do Bitcoin, onde informações não tão generalistas funcionam melhor. Para janela, foi escolhido novamente os últimos 3 dias, e essas novas colunas serão novamente baseadas em cada uma das colunas já existentes.

```
1. def momentos_estatisticos(self):
2.         for column in self.data.columns:
3.             if column != self.target:
4.                 #Média móvel e desvio padrão de 3 dias.
5.                 self.df_fe[f'{column}_moving_average'] =
   self.data[column].rolling(3).mean()
6.                 self.df_fe[f'{column}_std'] =
   self.data[column].rolling(3).std()
```

Combinações Polinomiais

As combinações polinomiais fazem operações entre diferentes features para buscar alguma nova feature que possua uma correlação boa. Novamente, trata-se de uma abordagem *data-driven*.

```
1. def combinacoes_polinomiais(self):
2.         df_poly = PolynomialFeatures(2)
3.         cols = self.data.columns
4.         df_poly = pd.DataFrame(df_poly.fit_transform(self.data[cols]))
5.         qtde_colunas = len(df_poly.columns)
6.         df_poly = df_poly.drop(columns=[x for x in range (len(cols)+1)])
7.         nome_novas_colunas = []
8.         nao_vistadas = list(cols.copy())
9.         for coluna in cols:
10.            atual = coluna
11.            nome_novas_colunas.append(f'{coluna}^2')
12.            for _ in nao_vistadas:
13.                if (_ != atual):
14.                    nome_novas_colunas.append(f'{coluna}*{_}')
15.
16.            nao_vistadas.remove(coluna)
17.
18.        nome_velhas_colunas = [x for x in range(len(cols)+1,qtde_colunas)]
19.        for i in range(nome_velhas_colunas[0],nome_velhas_colunas[-1]+1):
20.            df_poly =
   df_poly.rename(columns={i:nome_novas_colunas[i-nome_velhas_colunas[0]]})
21.
22.        for col in df_poly.columns:
23.            self.df_fe[col] = df_poly[col].values
```

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

Subtrações entre variáveis

Finalmente, para fechar a etapa de feature-engineering, foram feitas subtrações entre os valores base em diversas ordens:

- **high-low**: subtrai o valor de baixa (vale) do valor de alta (pico), gerando a amplitude de variação.
- **high-close**: subtrai o valor de fechamento do valor de alta.
- **low-close**: subtrai o valor de fechamento do valor de baixa.
- **close-open**: subtrai o valor de abertura do valor de fechamento.
- **high-open**: subtrai o valor de abertura do valor de alta.
- **low-open**: subtrai o valor de abertura do valor de baixa.

```
1.  def difference(self):
2.      df_final = pd.DataFrame()
3.      df_final['high-low'] = self.data['High'] - self.data['Low']
4.      df_final['high-close'] = self.data['High'] - self.data['Close']
5.      df_final['low-close'] = self.data['Low'] - self.data['Close']
6.      df_final['close-open'] = self.data['Close'] - self.data['Open']
7.      df_final['high-open'] = self.data['High'] - self.data['Open']
8.      df_final['low-open'] = self.data['Low'] - self.data['Open']
9.
10.     self.df_fe = self.df_fe.merge(df_final, left_index=True,
    right_index=True)
```

[5] - Modelo utilizado - XGBoost:

Foram realizados alguns testes com os seguintes modelos:

- *LSTM;*
- *MLP simples;*
- *XGBoost;*
- *Catboost.*

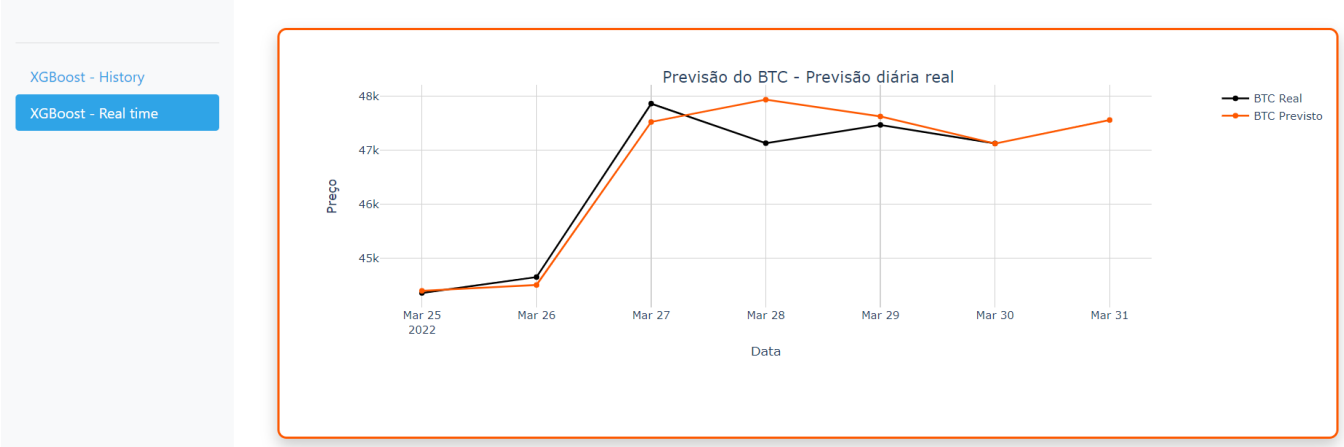
Entretanto, o algoritmo que apresentou melhores resultados foi o *XGBoost* com a seguinte configuração:

```
1.  class ModelXGboost():
2.      def __init__(self, X_train, y_train):
3.          self.X_train = X_train
4.          self.y_train = y_train
5.
6.
7.      def fit(self):
8.          self.model = XGBRegressor(n_estimators=1500, learning_rate=0.05,
    max_depth=12, random_state=42,
9.          eval_metric='mae', gamma=0.5, reg_lambda = 0.6, reg_alpha=0.7,
```

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

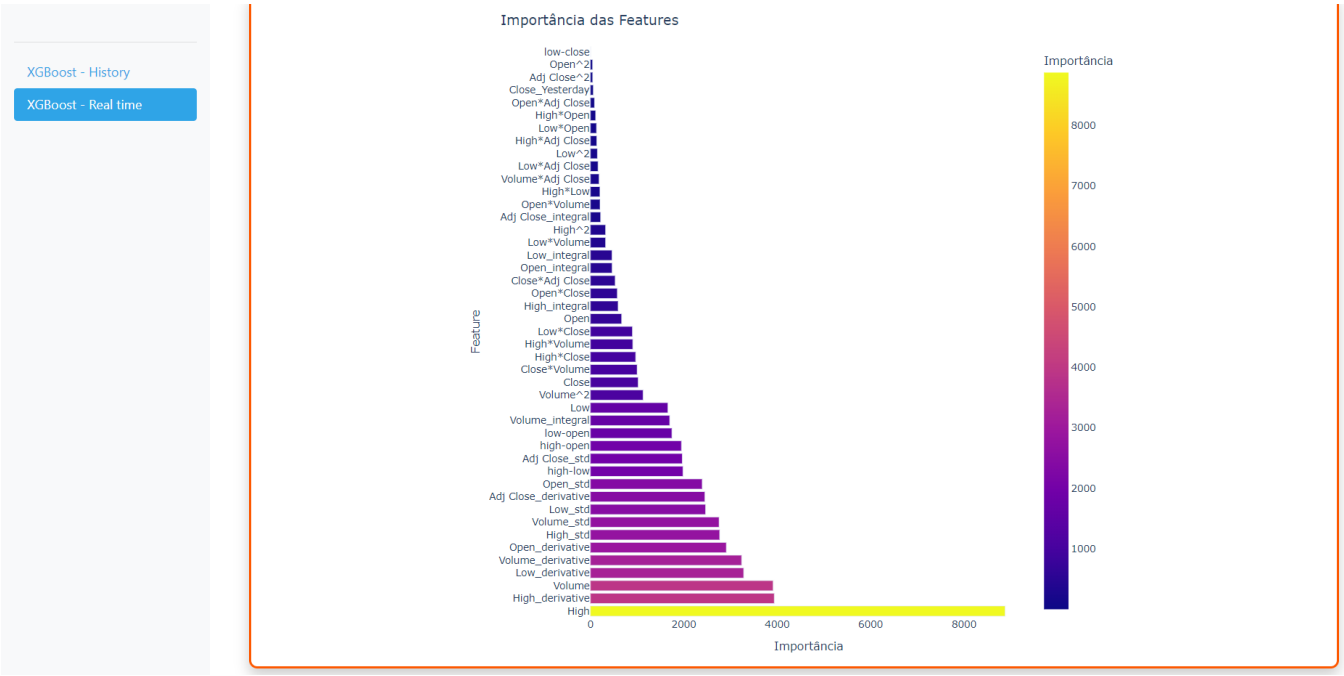
```
    subsample=0.8)
        self.model.fit(self.X_train, self.y_train)

    def predict(self, X_test_xgb):
        self.y_pred = self.model.predict(X_test_xgb)
        return self.y_pred

    def get_booster(self):
        return self.model.get_booster()

```

[6] - Resultados:



Universidade Federal de Viçosa

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*



Universidade Federal de Viçosa

Nome: João Victor Magalhães Souza. **Matrícula:** 3483.
Lucas Ranieri Oliveira Martins. **Matrícula:** 3479.
Disciplina: CCF 493 – Engenharia de Aprendizado de Máquina.
Professor: Dr. Fabrício Aguiar.
Relatório do projeto - *Bitcoin Estimator.*

Referências

[1] *XGBoost.* ***XGBoost 1.5.2 Documentation.*** Disponível em: https://xgboost.readthedocs.io/en/stable/index.html. Acesso em: 27/03/2022.

[2] *Dash.* ***Dash Documentation & User Guide*****.** Disponível em: https://dash.plotly.com/. Acesso em: 30/03/2022.