Relatório A2 - Linguagens de Programação

Victor Bombarda, Lucas Braga, Gabriel Machado e Eduardo Junqueira

8 de dezembro de 2020

1 Introdução

Este relatório relativo à A2 da disciplina Linguagens de Programação, ministrado pelo professor Rafael Pinho, foi elaborado por Victor Bombarda, Lucas Braga, Gabriel Machado e Eduardo Junqueira.

O trabalho foi desenvolvido utilizando Spyder (Anaconda), sendo usadas as seguintes bibliotecas de Python: Pandas, Sklearn, Seaborn, Matplotlib, Geopandas, Data Processing, Sphinx, Warnings e Descartes.

A divisão de tarefas ficou estabelecida como a seguinte:

- Eduardo Junqueira - Cientista de Dados / Especialista de Negócio
- Lucas Braga - Engenheiro de Dados / Engenheiro de Software
- Victor Bombarda - Especialista em Visualização de Dados
- Gabriel Machado - Especialista de Garantia da Qualidade

2 Limpeza dos dados

As tabelas escolhidas para serem aplicadas no trabalho foram as da Fifa e do Covid. Para a conexão dos dados, criamos classes que estabelecem o conexão com o banco. Dentro destas classes já foram filtradas as bases.

Na base da Fifa, foram "dropados"informações que consideramos inúteis para a análise pela qual propomos estudar a base. Assim, coisas como links para fotos dos jogadores, bandeiras do países, logos dos clubes foram desconsiderados.

Colunas com valores monetários também foram dropados pois muitas das informações eram especulações: contratos com jogadores em sua maioria são secretos, além do fato de haverem muitas observações vazias e não preenchidas.

Por último, jogadores que não tinham informações sobre altura ou peso, também foram dropados, pois, para responder as perguntas propostas por nossos integrantes, estas seriam informações cruciais.

Em relação aos dados da base da Covid, não houve muitas coisas necessárias para corrigir e tornar o dados minimamente corrigidos. Não observamos observações que houvessem valores vazios ou não preenchidos, porém percebemos que as colunas "AggregationMethod"e "Version"eram inúteis pois continham a mesma informação para todas as observações: era um erro na criação do banco.

Outra coisa importante que foi feita foi a transformação da data, que estava em "string", para "datetime"do próprio Pandas, assim, agora o programa consegue identificar a sucessão de eventos a partir das datas. Por último, extraímos as coordenadas latitude e longitude da coluna "Centroid", a partir do que era contido dentro do padrão POINT(...).

3 Dados Fifa

3.1 Briefing

A tabela da Fifa, tem diversas informações, nas quais cada linha corresponde a um jogador. Há aqui informações sobre o time desse jogador, a idade dele, o seu aproveitamento de gols, peso, altura e muitos outros detalhes que o definem como jogador e papel em campo.

3.2 Análise exploratória

```
# AN LISE EXPLORATRIA DOS DADOS DO FIFA
    # Quantidade de pa  ses presentes na fifa
fifa['Nationality'].nunique()

    # Quantidade de jogadores por pa  s
fifa['Nationality'].value_counts()

    # M dia de idades
fifa['Age'].mean()

    # Quantidade de jogadores por idade
fifa['Age'].value_counts()

    # M dia de idade por clube
fifa.groupby('Club').Age.mean()

    # M dia de sal rio por jogador
fifa['Wage'].mean()

    # M dia de sal rio por clube
fifa.groupby('Club').Wage.mean()

    # Gasto salarial por clube
fifa.groupby('Club').Wage.sum().sort_values(ascending = False)

    # Overall por idade
fifa.groupby('Age').Overall.agg(['min','max','mean'])

    # Potencial por idade
fifa.groupby('Age').Potential.agg(['min','max','mean'])

    # Melhores m dias de Overall por clube
fifa.groupby('Club').Overall.mean().sort_values(ascending = False)

    # Jogadores com mais finaliza  es
fifa.sort_values(by='Finishing',ascending = False)

    # Jogadores com maior pot ncia de chute
fifa.sort_values(by='ShotPower',ascending = False )

    # Jogadores mais r pidos
fifa.sort_values(by='SprintSpeed',ascending = False )
```

3.3 Pênaltis e Atributos

A primeira análise feita tabela foi relativa ao aproveitamento de pênalti de um time, no qual comparamos os 5 times com maior média deste aproveitamento com os 5 times com pior média neste quesito.

Ao analisarmos, encontramos os 10 times relativos à filtragem acima e criamos dataframes separados para cada uma. Elegemos então 4 fatores que aparentemente influiriam em um bom aproveitamento de pênaltis; são eles: finalização, curva, força e agressividade.

Antes de organizarmos os dados em ordem crescente, podemos observar que é difícil idenficar qualquer relação entre os dois dados, porém, com a organização, torna-se evidente o quão importante é cada atributo para o bom aproveitamento de pênaltis para um time.

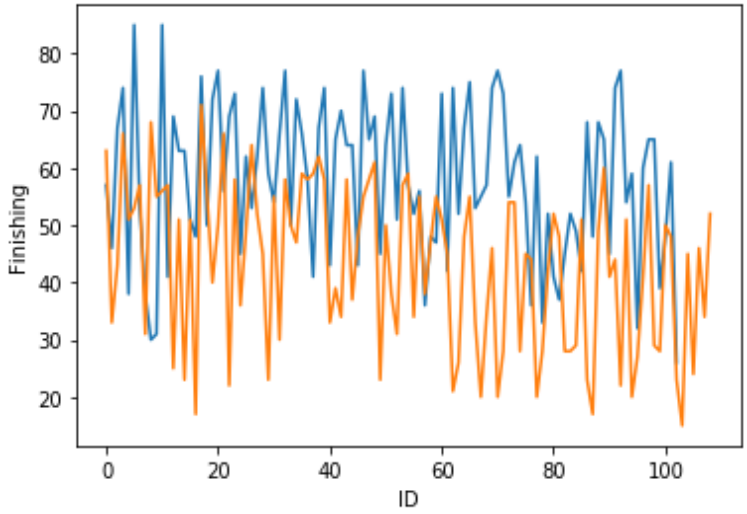


Figura 1: *Finalização desorganizada.*

A finalização tem a ver com a quantidade de vezes um jogador chuta ao gol. Na visualização podemos ver que os melhores times acambam tendo jogadores que mais tentam finalizações.

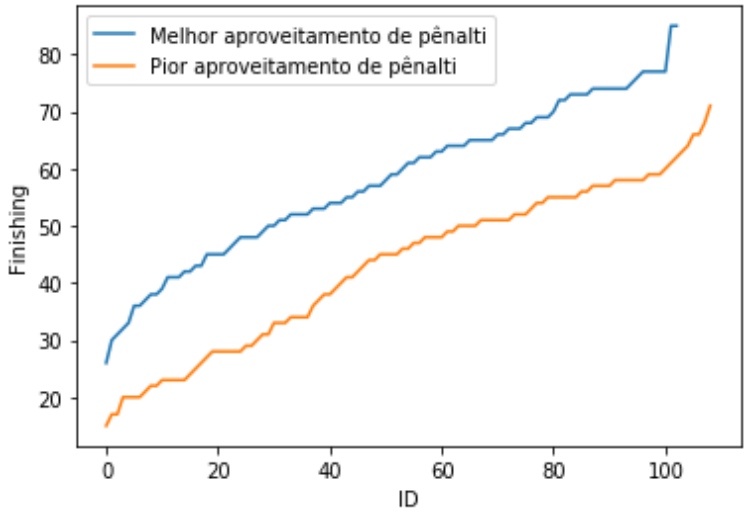


Figura 2: *Finalização.*

A curva retrata a capacidade de um jogador em dar um efeito à bola ao qual ela descreve uma trajetória quase que parabólica, dificultando o trabalho de um goleiro de defender um gol. Jogadores que melhor realizam esse efeito, mais tem chance de bater bons pênaltis.

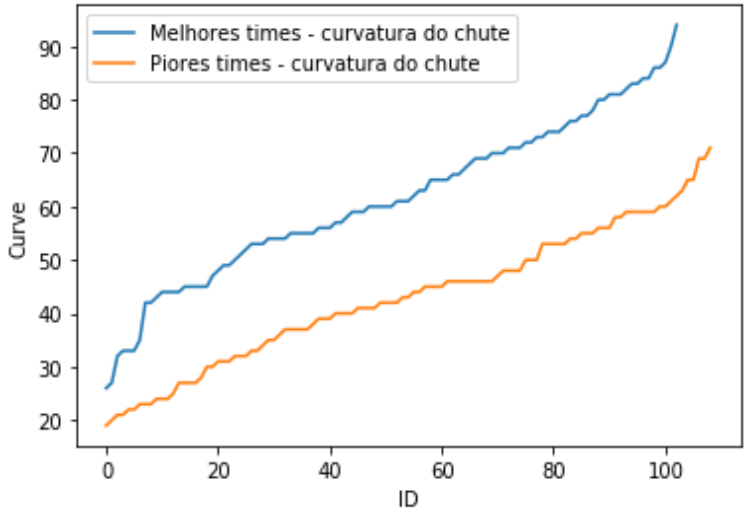


Figura 3: *Curvatura.*

O atributo força é relativo à força do chute de cada jogador, se ele tem mais capacidade de realizar longos chutes ou se foca em realizar mais passes e, consequentemente, chutes mais curtos. Antes de vermos a visualização, é óbvio que um chute forte é um dos fatores elementares para um bom pênalti.

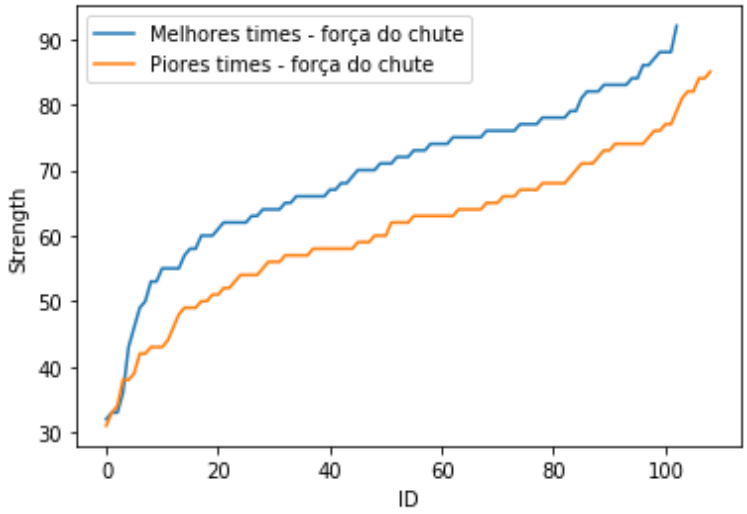


Figura 4: *Força do chute.*

O último atributo é a definição do estilo de jogo de um jogador, se ele possui uma postura mais passiva ou agressiva em relação ao eventos do jogo. Não é de impressionar que um time que bate boas faltas tem um elenco mais agressivo, como pode ser visto na imagem abaixo.

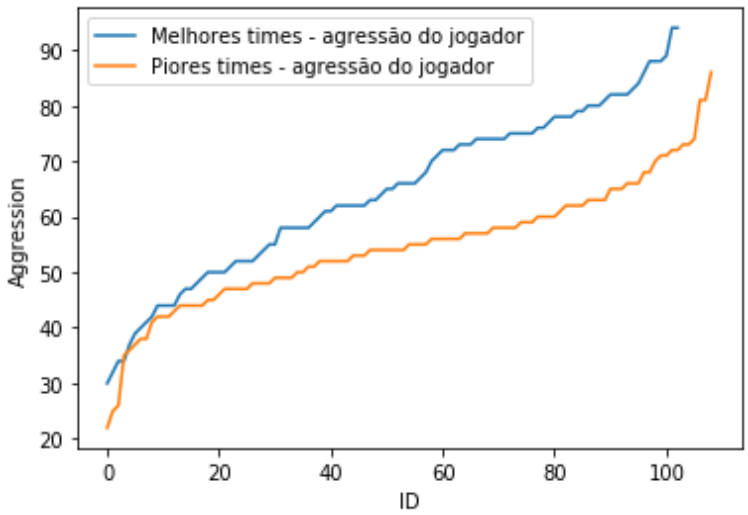


Figura 5: *Agressividade.*

3.4 Relação entre Idade e Qualidades

Nesta parte, agroupamos os jogadores por idade, e agregamos as observações mínima, máxima e a média relativa a cada atributo usado para comparação. Assim, podemos analisar possíveis relações existentes entre a idade de um jogador e suas qualidades, como o quanto pula, o quanto corre e coisas afins.

A primeira análise é referente ao salário. Com base no gráfico e em fatos do mundo do futebol, podemos ver que o ápice da carreira de um jogador de futebol se encontra em torno dos 25-30 anos de idade, aonde após este período, vemos que a média salarial tende a cair. Veremos nos gráficos subsequentes que jogadores com idades mais avançadas tendem a ter menos áptidão física do que os mais jovens, considerando que é um jogo de alto rendimento e que muito exige de seus participantes, jogadores profissionais conseguem manter um bom rendimento do seu máximo físico apenas durante alguns poucos anos.

- Código da Relação Média Salarial - Idade:

```
# SALARIO
SalarioPorIdade = fifa.groupby('Age').Wage.agg(["min","max","mean"])
print(SalarioPorIdade)
SalarioPorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig5 = sns.lineplot(x = SalarioPorIdade["Age"] , y =SalarioPorIdade["
    mean"],label = "M dia do sal rio por Idade", color = "red")
fig.savefig('imagens/fifa/analise_idade/Salario_idade.png')
```

- Plot:

Figura 6: *Relação Média Salarial - Idade.*

A segunda análise é da relação entre média geral, encontrada na tabela como "Overall", e idade dos jogadores. Essa média geral refere-se à nota atribuída a cada jogador considerando seu desempenho em geral. Apesar de termos alguns óbvios outliers jovens, como Mbappé (de 21 anos com Overall de 88), a média é relativamente baixa pois há mais jogadores, tendo menos jogadores com idades mais altas, sendo estes mais experientes. Assim, justifica-se que jogadores mais velhos tem uma nota relativamente mais alta.

- Código da Relação Nota Geral - Idade:

```
#NOTA GERAL
RatingPorIdade = fifa.groupby('Age').Overall.agg(['min','max','mean'])
RatingPorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig6 = sns.lineplot(x = RatingPorIdade["Age"] , y =RatingPorIdade["
    mean"],label = "M dia da nota geral por Idade", color = "blue" )
fig.savefig('imagens/fifa/analise_idade/NotaGeral_idade.png')
```

- Plot:

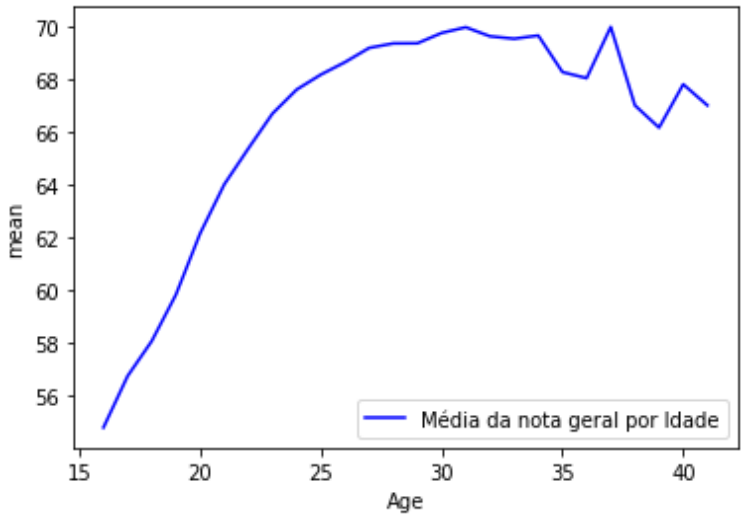


Figura 7: *Relação Nota Geral - Idade.*

A terceira análise é uma que requer poucos comentários, pois a imagem fala muito por si só. Ela compara a condição física dos jogadores em relação à idade. E acontece o esperado: com o tempo, o fólego dos jogadores vai acabando. Devido ao alto desempenho, muitos jogadores chegando ao fim dos 30 anos com lesões e costumeiramente fazem uma transição para virarem técnicos, onde aplicam seus conhecimentos de campo em treinos com jogadores mais jovens.

- Código da Relação Stamina - Idade:

```
#STAMINA
StaminaPorIdade = fifa.groupby('Age').Stamina.agg(['min','max','mean'
    ])
StaminaPorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig7 = sns.lineplot(x = StaminaPorIdade["Age"] , y =StaminaPorIdade[
    "mean"],label = "Stamina por Idade", color = "green" )
fig.savefig('imagens/fifa/analise_idade/Stamina_idade.png')
```

- Plot:

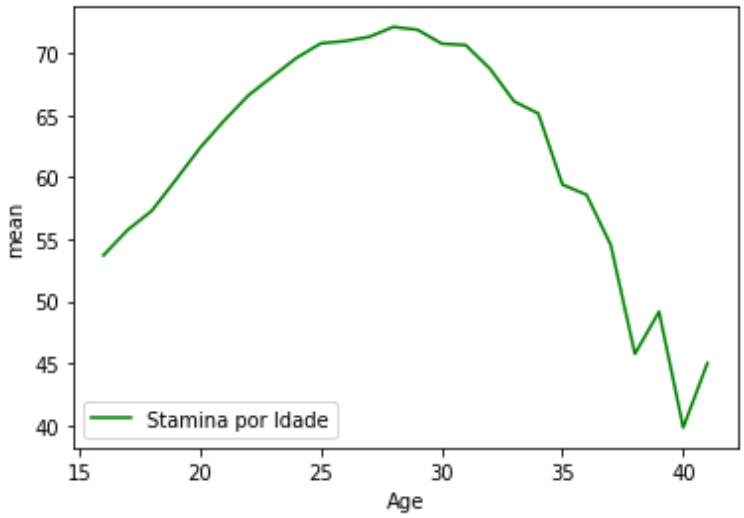


Figura 8: *Relação Stamina - Idade.*

A quarta análise segue a mesma ideia da anterior: condicionamento físico. Aqui vemos a relação do aproveitamento do pulo pela idade.

- Código da Relação Pulo - Idade:

```
#JUMPING - PULO
PuloPorIdade = fifa.groupby('Age').Jumping.agg(['min','max','mean'])
PuloPorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig8 = sns.lineplot(x = PuloPorIdade["Age"] , y =PuloPorIdade["mean"],
    label = "Pulo por Idade", color = "black" )
fig.savefig('imagens/fifa/analise_idade/Pulo_idade.png')
```

- Plot:

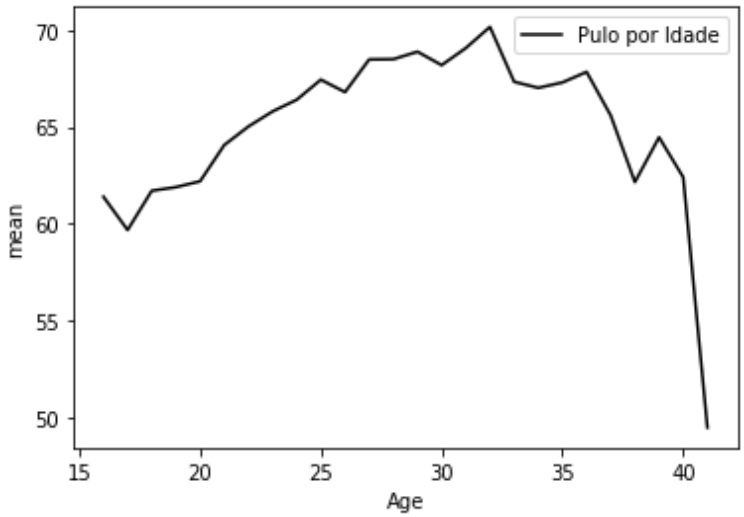


Figura 9: *Relação Pulo - Idade.*

Para a quinta análise, segue a agilidade relacionado com a idade. Podemos perceber que após os 40 anos temos um outlier (há apenas dois jogadores acima de 40 anos) que faz com que no final do gráfico haja uma subida na agilidade, não respeitando a regra observada entre os outros jogadores após o seu ápice: maior a idade, menor a agilidade.

- Código da Relação Agilidade - Idade:

```
#AGILITY - AGILIDADE
AgilidadePorIdade = fifa.groupby('Age').Agility.agg(['min','max','mean'])
AgilidadePorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig9 = sns.lineplot(x = AgilidadePorIdade["Age"] , y = AgilidadePorIdade["mean"],label = "Agilidade por Idade", color = "purple" )
fig.savefig('imagens/fifa/analise_idade/Agilidade_idade.png')
```

- Plot:

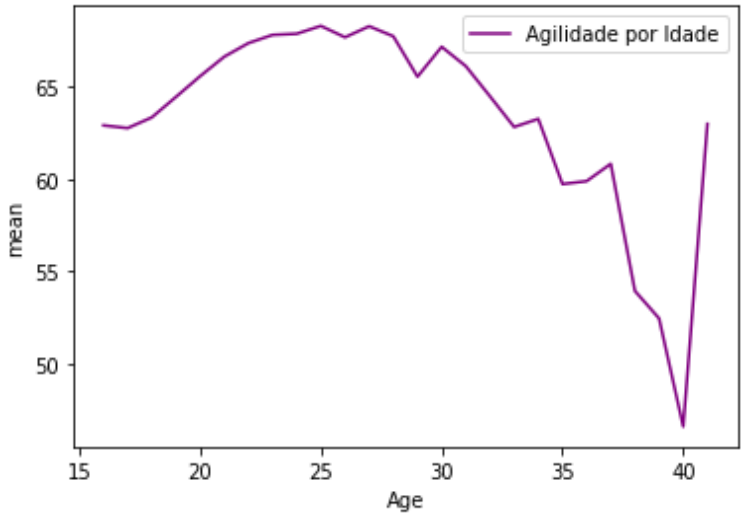


Figura 10: *Relação Agilidade - Idade.*

E por útlima analise temos uma análise de outro atributo físico: média de peso e idade.

- Código da Relação Peso - Idade:

```
#WEIGHT - PESO
PesoPorIdade = fifa.groupby('Age').Weight.agg(['min','max','mean'])
PesoPorIdade["Age"] = range(16,42)
fig = plt.figure()
ax = plt.subplot(111)
fig10 =sns.lineplot(x = PesoPorIdade["Age"] , y =PesoPorIdade["mean"],
    label = "Peso por Idade", color = "cyan" )
fig.savefig('imagens/fifa/analise_idade/Peso_idade.png')
```

- Plot:

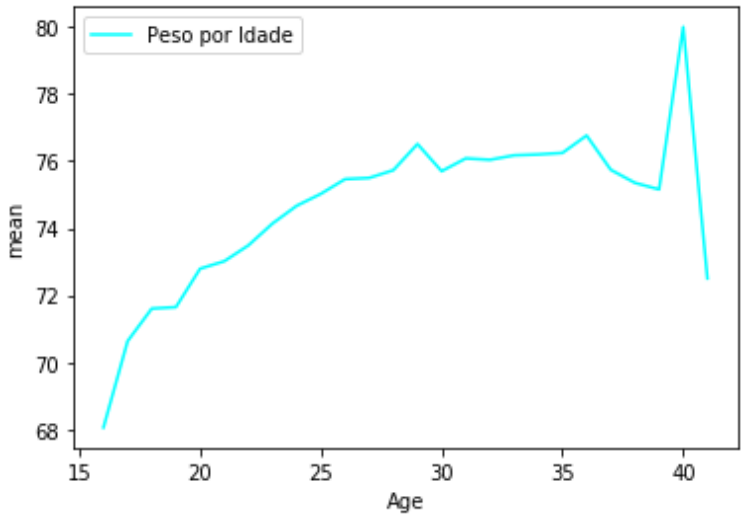


Figura 11: *Relação Peso - Idade.*

3.5 Regressão Linear - Fifa

A regressão linear foi realizada utilizando as ferramentas da biblioteca de Python Sklearn.

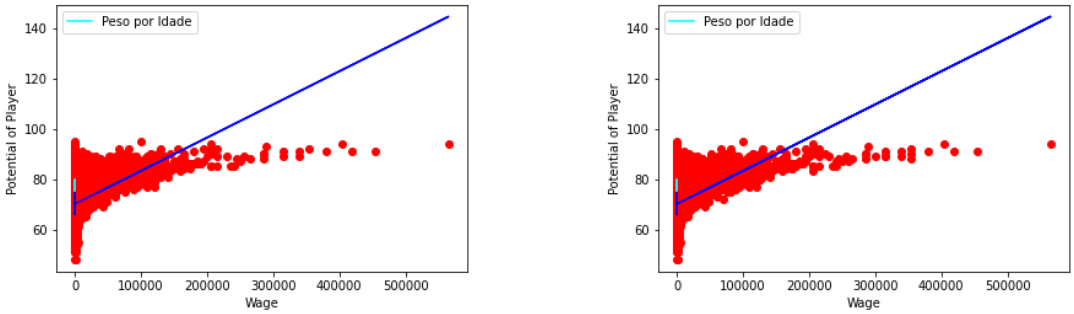
A primeira regressão, feita primeiro o treino, depois o teste. Nela foi calculada a relação entre a idade e o potencial, itens presentes na tabela. Podemos perceber uma tendência de queda no potencial de um jogador ao passar dos anos. Como já discutido em outras visualizações, conforme um jogador fica mais velho, fica menos rápido e menos ágil, assim também fica evidente que seu potencial igualmente diminui.

- Código da Regressão Linear:

```
# Regress o linear ----------------------------------------------
# cria   o de um modelo para prever a rela   o entre potencial do
    jogador e sua idade
x=fifa["Age"] # Vari vel independente
y=fifa["Potential"] # Vari vel dependente
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,
    random_state=0)
model = LinearRegression()
x_train=np.array(x_train)
y_train=np.array(y_train)
x_train=x_train.reshape(-1,1)
y_train=y_train.reshape(-1,1)
model.fit(x_train,y_train)
x_test=np.array(x_test)
x_test=x_test.reshape(-1,1)
y_pred= model.predict(x_test)

    # Visualizando dataset de treino
    # Onde os dados ser o utilizados para refinir o sistema de
        predi   o
plt.scatter(x_train,y_train,color="red")
plt.xlabel("Age of Player")
plt.ylabel("Potential of Player")
fig = plt.Figure()
fig.set_canvas(plt.gcf().canvas)
ax = plt.subplot(111)
fig14 = plt.plot(x_train, model.predict(x_train),color="blue")
fig.savefig('imagens/fifa/regressoes/regressao_linear1_treino.png')

    # Visualizando dataset de teste
    # Onde os dados ser o previstos com base no conjunto de treino,
        medindo a efici ncia de previs o
plt.scatter(x_test,y_test,color="red")
plt.xlabel("Age of Player")
plt.ylabel("Potential of Player")
fig = plt.Figure()
fig.set_canvas(plt.gcf().canvas)
ax = plt.subplot(111)
fig15 = plt.plot(x_train, model.predict(x_train),color="blue")
fig.savefig('imagens/fifa/regressoes/regressao_linear1_teste.png')

    # Encontrando interse   o da linha de regress o
model.intercept_

    # Encontrando coeficiente da regress o linear
```

```
    #    o grau de afinidade entre as vari veis , definindo se est o
        muito relacionadas (1) ou n o possuem rela   o (0)
model.score(x_train,y_train)

    # Encontrando mse (mean squared error)
mse = metrics.mean_squared_error(y_test,y_pred)
print(mse)

    ## Podemos notar que n o h  uma rela   o entre o potencial do
        jogador e sua idade (atrav s do R2 - model.score()), o que nos
         leva a buscar outro dado que se adeque melhor.

    ## J  nesse modelo, podemos notar que h  rela   o positiva
        entre o potencial do jogador e seu sal rio (atrav s do R2 -
        model.score())
# Regress o linear - prevendo potencial baseado no sal rio
x=fifa["Wage"] # Vari vel independente
y=fifa["Potential"] # Vari vel dependente
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,
    random_state=0)
model = LinearRegression()
x_train=np.array(x_train)
y_train=np.array(y_train)
x_train=x_train.reshape(-1,1)
y_train=y_train.reshape(-1,1)
model.fit(x_train,y_train)
x_test=np.array(x_test)
x_test=x_test.reshape(-1,1)
y_pred= model.predict(x_test)

    # Visualizando dataset de treino
    # Onde os dados ser o utilizados para refinir o sistema de
        predi   o
plt.scatter(x_train,y_train,color="red")
plt.xlabel("Wage")
plt.ylabel("Potential of Player")
fig = plt.Figure()
fig.set_canvas(plt.gcf().canvas)
ax = plt.subplot(111)
fig16 = plt.plot(x_train, model.predict(x_train),color="blue")
fig.savefig('imagens/fifa/regressoes/regressao_linear2_treino.png')

    # Visualizando dataset de teste
    # Onde os dados ser o previstos com base no conjunto de treino,
        medindo a efici ncia de previs o
plt.scatter(x_test,y_test,color="red")
plt.xlabel("Wage")
plt.ylabel("Potential of Player")
fig = plt.Figure()
fig.set_canvas(plt.gcf().canvas)
ax = plt.subplot(111)
fig17 = plt.plot(x_train, model.predict(x_train),color="blue")
fig.savefig('imagens/fifa/regressoes/regressao_linear2_teste.png')
```

- Plots:



(a) Regressão treino.

(b) Regressão teste

Figura 12: As duas figuras que relacionam salário e idade de um jogador.

O mesmo foi feito relacionando salário e o potencial do jogador. Neste caso, podemos observar uma tendência de crescimento, onde quão maior o potencial de um jogador, maior será a tendência por um salário maior.

Para a regressão múltipla, foram usadas como variáveis independentes idade, nota geral (overall) e potencial. A variável dependente analisada foi o salário.

- Código da Regressão Múltipla:

```
# Encontrando interse  o da linha de regress o
model.intercept_

    # Encontrando coeficiente da regress o linear
    #    o grau de afinidade entre as vari veis, definindo se est o
        muito relacionadas (1) ou n o possuem rela  o (0)
model.score(x_train, y_train)

    # Encontrando mse (mean squared error)
mse = metrics.mean_squared_error(y_test, y_pred)
print(mse)

    ## Foram utilizados dados para definir uma correla  o entre o
        sal rio dos jogadores e sua idade, pontua  o de overall e
        potencial
# Regress o m ltipla - sal rio por idade, overall e potencial
x=fifa[["Age","Overall","Potential"]]
y=fifa["Wage"]

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
model = LinearRegression()
model.fit(x_train, y_train)
model.predict(x_test)
y_pred= model.predict(x_test)

    # Visualizando valores atuais e previstos de sal rio por jogador
fig = plt.Figure()
fig.set_canvas(plt.gcf().canvas)
ax = plt.subplot(111)
fig18 = plt.scatter(y_test,y_pred), plt.xlabel("Actual Wage"), plt.
    ylabel("Predicted Wage")
fig.savefig('imagens/fifa/regressoes/regressao_multipla_teste.png')
```

```
30      # Afinidade entre os dados avaliados
        # H  uma afinidade de 0.35 entre o sal rio dos jogadores e os
            demais dados avaliados
32  model.score(x_train, y_train)
```

- Plot:

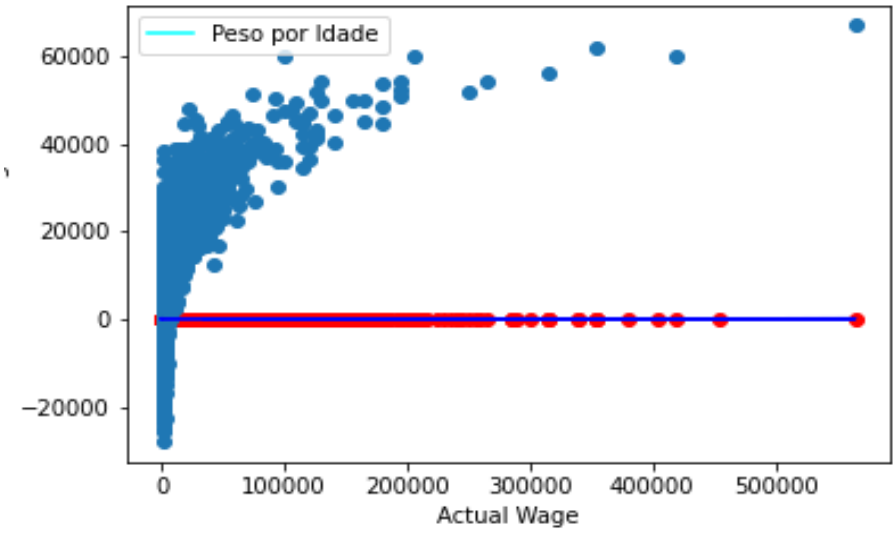


Figura 13: *Regressão múltipla.*

3.6 Tabela de correlação

Para última exposição dos dados desta tabela, faremos um gráfico que mostra a correlação entre duas colunas da tabela. O quão mais escuro estiver, maior a correlação, quanto mais claro, menor a correlação.

- Código do HeatMap de correlação:

```
## CRIANDO matriz de correla  o e gr fico heatmap
# Avaliando correla  o entre atributos
analise = fifa.filter(['Age','Overall', 'Potential','Wage', 'Agility',
    'Finishing', 'Acceleration','Ball control','Free kick accuracy','
   Jumping', 'Long passing' ])
# Matriz de correla  o
corr = analise.corr()
# Plotando matriz de correla  o
fig = plt.figure()
ax = sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=
    corr.columns.values,linewidths=0.25, vmax=1.0, square=True, cmap =
    'PuBu', linecolor='black', annot=False)
fig.savefig('imagens/fifa/aprov_penaltis/tab_correlacao.png')
```

- Plot:

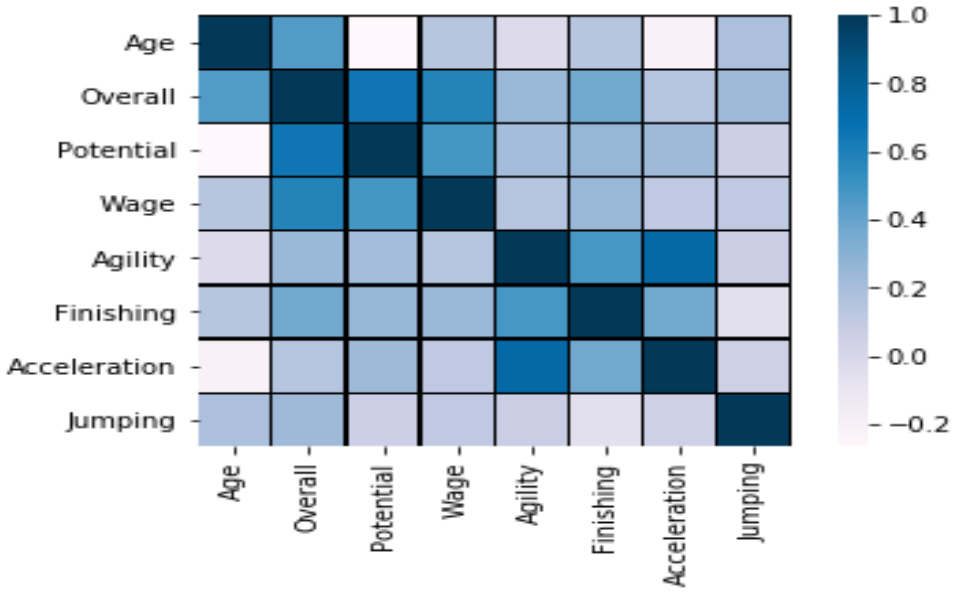


Figura 14: *Tabela de correlação.*

4 Dados Covid

4.1 Briefing

Sobre os dados da Covid, havia menos informações sobre as quais fazer perguntas e elaborar relações entre os dados presentes. Entretanto, trabalhos com a base e desenvolvemos algumas visualizações com o pouco de dado númerico que temos acesso: PercentOfBaseline. Com essa porcentagem do número de pessoa que passam pelo aeroporto, pudemos influir algumas coisas e relacionar com a passagem do tempo.

Um dos pontos mais importantes do ano é o próprio tempo em si: estamos em uma pandemia que, a cada dia que se passa, fica mais próxima do seu fim. Assim, desde o início do ano, houve um pico, um ápice em que o maior número de pessoas ficou em casa.

Com o passar dos meses, devido à pressões de necessidade econômica e humanitária, as regras que estabeleceram a quarentena se tornaram cada vez mais brandas, permitindo cada vez mais o fluxo de pessoas, o funcionamento de atividades não essenciais, como, por exemplo, aeroportos.

4.2 Procedência e mapa

A primeira visualização é em relação aos dados coletados: qual a procedência geográfica das informações. Utilizando uma biblioteca de Python que encontramos na internet, pudemos plotar com pontos em um mapa a localização dos aeroportos, utilizando as informações de coordenadas presentes na base. Apesar delas estarem "sujas", no meio de outras "strings"e mesmo depois de adquiridas ainda precisavam ser transformadas, foram de suma importância para a compreensão dessas informações adquiridas da tabela.

- Código do Mapa dos dados da tabela:

```
## COVID ------------------------------------------------
covid["Lat"] = ""
covid["Long"] = "" #Separa a Latitude da Longitude do arquivo original
    fornecido
for i in range(len(covid["Centroid"])):
    covid["Centroid"][i] = covid["Centroid"][i][6:-1]
    covid["Lat"][i] = covid["Centroid"][i].split(" ")[0]
    covid["Long"][i] = covid["Centroid"][i].split(" ")[1]

#Achamos no mapa o lugar dos aeroportos da basex
gdf = gpd.GeoDataFrame(covid, geometry=gpd.points_from_xy(covid["Lat"
    ],covid["Long"]))
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
ax = world.plot(column='PercentOfBaseline',color='white', edgecolor='
    black',figsize=(15,9))
ax.set_title('Aeroportos analisados na tabela - Covid', fontdict= {'
    fontsize':25})
ax.set_axis_off()
ax.get_figure()
gdf.plot(ax=ax, color='red')
plt.savefig("imagens/Covid/mapa .png")
plt.show() #Printa cada aeroporto como uma bolinha vermelha no mapa
```

- Plot:

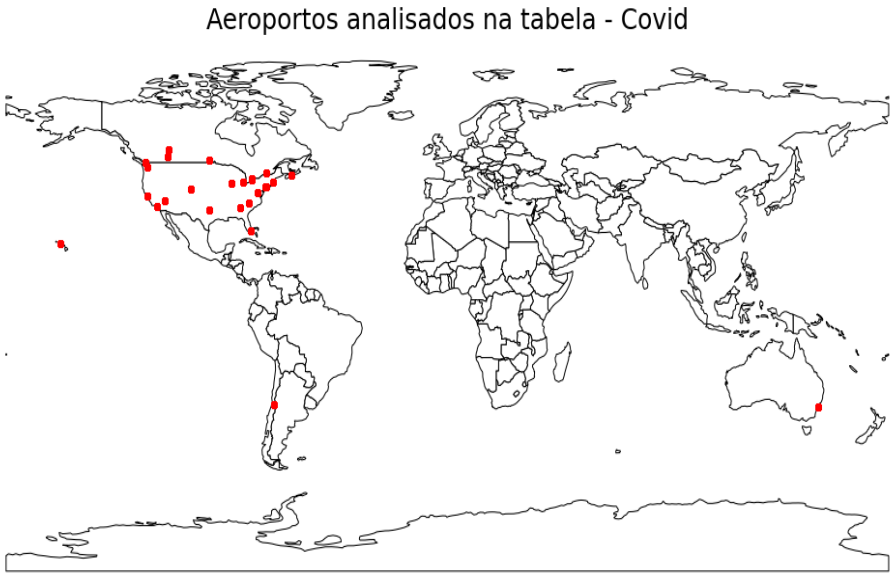


Figura 15: *Mapa mostrando em pontos de onde veio a informação.*

4.3 Análise dos fluxos - Covid

Em uma das primeiras visualizações, temos o fluxo nos aeroportos (estimado pelo PercentOfBaseline) ao longo do ano de 2020, no qual podemos analisar separadamente como ficou o fluxo de pessoas nos aeroportos de cada país. Interessante é observar que o Canada aparece em primeiro lugar. Poderíamos dizer que, pelos EUA terem tido uma queda não muito brusca no fluxo, apresenta-se umas das consequências dos EUA ser o país com maior número de casos e mortes no mundo. Entretanto, o Canada torna impossível concluir isto, tendo em vista que o Canada foi afetado de forma muito mais branda que os EUA e mesmo assim teve um aumento deste fluxo logo em julho, pico da epidemia no Brasil, por exemplo.

- Código do Fluxo nos aeroportos:

```
## IMPACTO DO COVID 19 NOS AEROPORTOS:

covid = dfc().fetcher()
covid.head()
# Percent of Baseline    a propor  o de viagens comparado com a
    m dia de viagens feitas no mesmo dia da semana no per odo base.

# Checando informa  es da tabela
covid.info()
covid.describe()
covid['Country'].unique()
# Organizando as datas de acordo com os meses em que foram registradas
covid['Day'] = pd.DatetimeIndex(covid['Date']).dayofyear
covid['Day'].value_counts()
covid['Month'] = pd.DatetimeIndex(covid['Date']).month
covid['Month'].value_counts()

# Divis o das informa  es geogr ficas:
```

```
    # Contagem por pa ses
covid.Country.value_counts()

    # Contagem por estados
covid.State.value_counts()

    # Contagem por cidades
covid.City.value_counts()

# Visualiza  o da varia  o do tr fego ao longo dos meses de 2020
fig = plt.Figure(dpi = 300)
#fig.set_canvas(plt.gcf().canvas)
ax1  = plt.plot(121)
fig20 = sns.lineplot(x=covid['Month'], y=covid['PercentOfBaseline'],
    hue="Country", data=covid)
plt.tight_layout(pad=0.01)
plt.show()
fig.savefig('imagens/covid/trafego_por_mes.png')
```

- Plot:

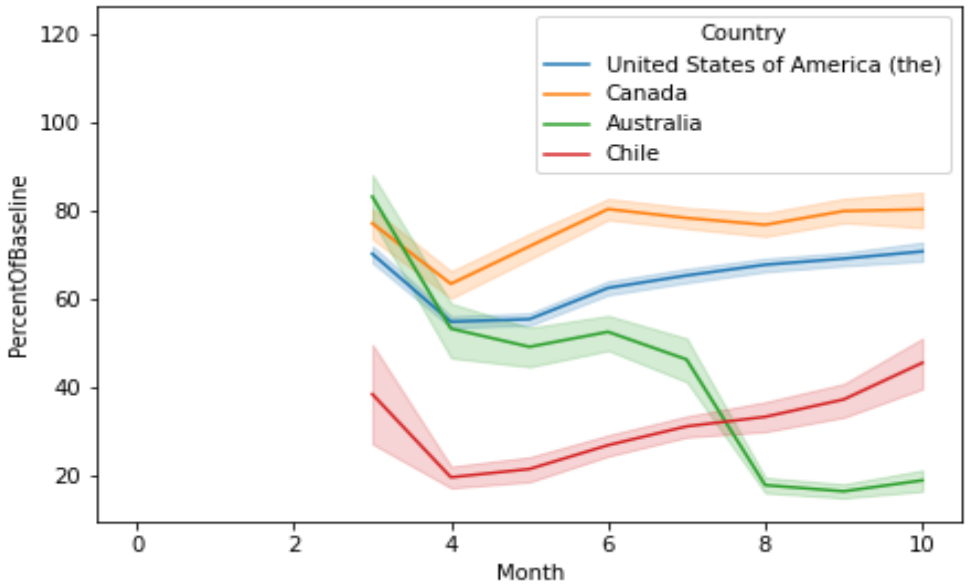


Figura 16: *Fluxo nos aeroportos, por país, ao longo do ano.*

Outra visualização que se prende, basicamente, no mesmo conceito é a próxima: boxplot dos fluxos por país. Agora podemos ter uma análise um pouco mais sutíl se combinada com conhecimento político estrangeiro e sobre a conjuntura dos EUA nesse ano específico(eleição presidencial). Aqui podemos ver que a discrepância nos EUA em relação ao número máximo e mínimo de fluxo que encontramos nas amostras é extremamente alto. Podemos ver que alguns números sequer foram computados no boxplot pois o programa idenficou eles como outliers (um número, inclusive, expressivo). O que podemos perceber é que, como nos EUA os estados possuem mais autonomia em relação ao governo federal, logo cada estado lutou sua própria guerra contra o Covid, alguns mais bruscamente, outros de forma mais branda. Mas o fato é, considerando o posicionamento do atual presidente dos EUA e de sua oposição, conseguimos entender que o que faz essa descrepância ter este tamanho. O fato comentado acima sobre o Canada continua sendo confirmado aqui.

- Código do Boxplot do Fluxo em cada país:

```
# Visualiza  o em boxplot do grau de impacto do covid por pa s
fig = plt.Figure(dpi = 300)
#fig.set_canvas(plt.gcf().canvas)
ax1  = plt.plot(121)
fig21 = sns.boxplot(x=covid['Country'], y=covid['PercentOfBaseline'],
    palette=["m", "g"], data=covid)
fig.savefig('imagens/covid/trafego_por_pais.png')
```

- Plot:

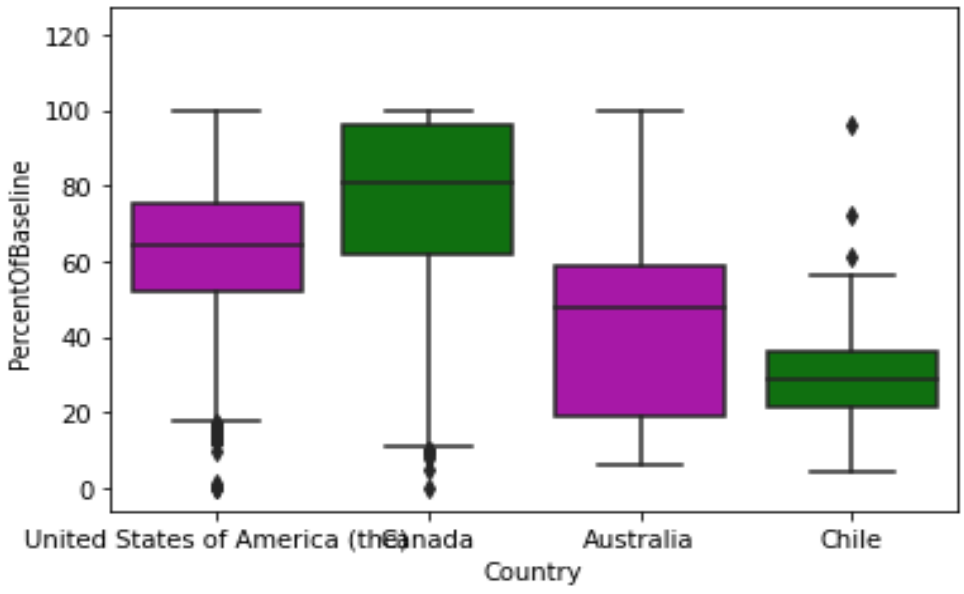


Figura 17: *Boxplot do fluxo nos aeroportos em cada país.*