Materiais:

Q: Onde encontro os materiais de apoio e os arquivos notebooks para execução dos exercícios?

R: O material de apoio consta no ícone de folha a4 ao lado do título da disciplina, os notebooks do professor André e do professor Juliano estão em uma página do Dropbox, que o estudante poderá fazer o download e executar.

Os notebooks do professor André são os nomeados como (Arquivos YPNB).

Durante a aula 01, o professor André indica que o conjunto de dados Municípios estará nos slides, mas não consta. Mas os notebooks estão disponíveis na folha a4 ao lado do nome da disciplina e ao abrir o notebook ele possui o link e a solução proposta.

Ferramentas:

Q: Gostaria de saber qual é a maneira (linhas de código) de importar um arquivo csv direto do meu computador para o Colab?

R: Diretamente do desktop não é possível ler os arquivos, mas se há acesso ao Google Driveé possível fazer o upload desses arquivos no google drive e então, ler no Google Colab a partir do Google Drive. O Colab oferece uma biblioteca para isso, abaixo um exemplo:

```
from google.colab import drive

drive.mount('/content/drive', force_remount=True)

file_path = "/content/drive/My Drive/new_data_full.csv"

data = pd.read_csv(file_path, sep=';').replace({np.NaN: None})
```

Obs.: Ele irá pedir que você faça o login e dê permissão para acesso ao seu Google Drive.

Obs 2.: Sempre que reiniciar o kernel você terá que fazer isso.

Q: Referente à área de tratamento de textos, como fazer para extrair informações de arquivos PDF, arquivos texto etc. Como transformar esses textos em formatos semiestruturados pelo menos, como CSV?

R: Para soluções de problemas mais simples, é possível usar uma lib como o "tabula".

Segue referências que podem ajudar:

https://tabula-py.readthedocs.io/en/latest/getting_started.html#example

https://betterprogramming.pub/convert-tables-from-pdfs-to-pandas-with-python-d74f8ac31dc2

Para soluções mais robustas pode ser que seja necessária uma plataforma específica para converter e armazenar textos, como por exemplo o elastic search Solr. Depois basta acessar o Solr via Python para fazer os trabalhos de ciência de dados.

Conceitos e Exercícios:

Q: Na resolução da parte 3 do exercício 2 (vídeo parte 2, aprox. 18min) fiquei em dúvida em relação a como foi feita a atribuição da média para as extremidades. Como foi aplicado em duas etapas, a média atribuída para cada extremidade será diferente.

R: Sobre a diferença, ela seria menor que 0.1 numa escala de 100. Na maior parte do caso, não vai fazer diferença no resultado, exploração ou ML. No entanto, o método correto realmente seria substituir ambas as extremidades na mesma instrução, além de ser a maneira mais elegante.

Q: Sobre o exercício a respeito do time que mais marcou gols:

R: A maneira que o professor Juliano agrupou os times considera qual time marcou mais gols como mandante e não ao todo, diferente do que o exercício propôs. O professor Juliano encontrou o Santos como goleador daquela temporada, enquanto na verdade o melhor ataque foi do Corinthians (Campeonato Brasileiro de Futebol de 2015 – Wikipédia, a enciclopédia livre (wikipedia.org)).

Para chegar à resposta correta, devemos somar os gols que cada clube marcou como mandante e como visitante para encontrar quem de fato marcou mais gols.

Q: Sobre o exercício e o conjunto de dados da NBA:

R: Ao pesquisarmos sobre o jogador Wilt Charmberlain no Google é possível descobri que a temporada em que ele mais marcou foi a de 1962, com 4029 pontos listados. Já em 1965, ele marcou na verdade 2534 pontos, metade do que está no banco de dados.

Em 1965 ele trocou de time, e o banco reflete a pontuação dele em cada time (1480 + 1054) e a pontuação total do ano (2534), e é por isso que a soma no código resulta no dobro do valor real de pontuação.

Outro problema no banco de dados é que ele indica que Eddie Johnson possui o maior número de temporadas jogadas, com 33 temporadas. Até a data de fechamento do banco de dados, deveríamos encontrar Kevin Garnett e Kevin Willis, ambos com 21 temporadas.

Como há mudança de clube por parte dos jogadores no meio da temporada, há múltiplas entradas para cada jogador em cada ano, então deveríamos ter filtrado por valores únicos (. nunique()). Ainda assim, teríamos Mike Dunleavy com 26 anos

incorretamente em primeiro lugar. Ao pesquisarmos novamente é possível descobrir que é porque Mike Dunleavy pai e filho, ambos de mesmo nome, jogaram na NBA, porém o banco registra apenas o pai.

Q: Levando em consideração as análises de erros em dados acima, o que podemos fazer para evitar que a gente cometa esses erros em análises do mundo real no dia a dia? Como evitar cair nessas possíveis armadilhas dos dados?

R: Na verdade no mundo real nem sempre temos a resposta real tão fácil assim para compararmos as bases de dados, não é? As vezes os dados são agregados, sumarizados e transformados até a base que está sendo utilizada para a construção de alguma transformação.

Nos casos reais, geralmente as bases de dados são alvo de uma boa análise exploratória antes mesmo de construir qualquer tipo de solução ou algo do tipo. Existem analistas de dados, analistas de negócio também o próprio cientista de dados que faz essa análise utilizando métricas, explorando os max, min, média, mediana entre outros.

Acho que no caso peculiar da base de dados do NBA, também se encaixa técnicas de detecção de outliers, visto que quando verificaste que o Eddie Johnson possuía 33 temporadas é algo meio destoante dos outros, visto que ele necessitaria começar a jogar talvez com 9 anos rsrsrs.

Sobre a questão dos pontos do Wilt, também considero como erro de quem está preenchendo ou até mesmo um quesito que foi considerado na hora de preencher, nesse caso, teríamos que ter um conhecimento prévio da construção do conjunto de dados, que também, nem sempre temos a disposição.

Todas as bases de dados estão sujeitas a ter esse tipo de erro, e é importante a etapa de análise exploratória justamente para identificá-los e verificar como é possível ajustá-los ou até mesmo excluir esses registros.

Q: Ao fazer todo o exercício da aula, percebi que eu estava com muito mais colunas, do que o esperado. Executando o comando info(), percebi que havia várias colunas com nomes iguais mudando apenas o numeral no fim (e.g. desc_clube1 e desc_clube2). O que pode ter acontecido?

R: Se executarmos mais de uma vez cada trecho de código (célula), vai acabar criando outras colunas de dados (*features*). Então, quando estiver com o programa pronto, reinicie o kernel do seu Jupyter ou Colab e execute tudo de uma vez, assim garante que vai executar cada célula apenas uma vez para obter as colunas sem redundância.

Q: Para o valor médio do FTr, acredito que precise ainda de um ajuste pois como se trata de uma taxa, a média não deveria ser maior do que 1.

R: Para o FTr, é possível fazer o ajuste pra ser com base no valor e não em quantile(). Usando como regra o intervalo]0,1[.

Q: Em diversos momentos foi falado sobre padronização dos dados. Em um dos momentos até foi trazido a questão do z-score como forma de padronização. A minha dúvida é mais conceitual: o que é a padronização dos dados? Por que ela é feita e como ela pode ser feita?

R: Quanto ao conceito de padronização dos dados, é uma questão de deixá-los em uma mesma escala, pense que temos variáveis em diferentes unidades, variável A decimal de 0 a 1, variável B de 0 a 1000 entre outros exemplos, dessa maneira, o modelo que você está construindo, pode ter um viés para considerar variáveis com os valores mais altos ou para os menores, dada essa diferença de valores. Dessa maneira há maneiras de padronizar como o z-score onde se subtrai de cada variável sua média, e depois se divide o resultado pelo seu desvio padrão, o min-max entre outras que deixam todos esses valores em uma mesma escala. Um exemplo da min max pode ser visto aqui também: https://learn.64bitdragon.com/articles/computer-science/data-processing/min-max-normalization#:~:text=Min-max%20normalization%20is%20an%20operation%20which%20rescales%20a,also%20called%20the%20lower%20bound%20or%20least%20element

Q: A normalização é uma aproximação dos dados de uma curva normal. E aí fiquei com uma dúvida: como devemos proceder quando os dados não seguem uma distribuição normal? Se eles seguirem uma distribuição de Weibull, por exemplo, como "estandizar" esses dados?

R: A normalização vai fazer com que os dados estejam mais homogêneos e em escalas mais justas entre eles (e.g. variável a na casa das dezenas, enquanto variável b na casa das milhares). Existem algumas técnicas como z-score e min-max score que irão fazer esse tipo de normalização.

Q: No caso, estamos falando de um pequeno volume de dados das reportagens sobre turismo e já começou a ficar um pouco lento. Se estivéssemos analisando uma quantidade maior de informação, qual seria a abordagem para trabalhar com esses dados? A segunda pergunta seria sobre Análise de Sentimento. Há alguma biblioteca como a spacy mas voltada para verificar se um determinado texto é positivo/negativo/neutro? Procurei bastante mas não achei nada em português.

R: Quanto maior a quantidade de dados tratados e quanto mais etapas de processamento são necessárias no pipeline, maior a necessidade de recursos (memória, processador e etc..) para processar. Assim, alternativas como computação em nuvem são mais indicadas, para alguns pipelines ainda é possível utilizar o Google Colab ou o Amazon SageMaker. Mas se mesmo assim ficar muito pesado o processamento, alternativas como clusters na nuvem são o mais indicado, e aí é possível utilizarmos Spark com PySpark, R, Scala entre outros. Quanto a bibliotecas para análise de sentimento, não é do meu conhecimento bibliotecas prontas para isso, mas sim algumas que podem te auxiliar na construção de modelos de classificação para posteriormente verificar se o texto é positivo/negativo/neutro. Usando o seu exemplo de biblioteca, deixo dois tutoriais para inspiração do amigo: Natural Language Processing With spaCy in Python – Real Python e Sentiment Analysis with Spacy and Scikit-Learn | Engineering Education (EngEd) Program | Section.

Q: Professor, em relação ao PCA, para utilizar o componente em um modelo de regressão, por exemplo, a variável independente será o peso das componentes? Como aplicá-lo?

R: Para utilizar a redução de dimensionalidade com o objetivo de um modelo de regressão, é utilizado uma ramificação da PCA chamado PCR: PCR é uma regressão linear entre a variável Y e o novo sistema “reduzido” de variáveis X (definido pelos componentes principais).

Q: Não consegui identificar uma diferença entre as funções Merge e Join, aparentemente fazem a mesma coisa

R: São funções bem parecidas mas possuem suas particularidades ao fazer a junção entre dois dataframes. pandas.merge() é a função subjacente usada para todos os comportamentos de merge/join. Por padrão, a função Join do Pandas funcionará na junção através do índice dos Data Frames e podemos fazer com que A diferença básica entre a operação de merge e join vem da chave ou de um código comum que foi usado pelas duas operações. Para join sempre que damos um comando como df1.join(df2) a junção ocorre no nível de índice de df2. O índice será a chave para unir os Data Frames. Já no Merge quando damos um comando como pd.merge(df1, df2) ele procura por colunas comuns nos Data Frames df1 e df2 e podemos juntá-lo com uma ou mais colunas.Por outro lado, o merge na junção padrão por meio de uma coluna de Data Frames e podemos fazer com que ela execute a junção do índice dandoPd.merge (df1, df2, right_index=True)

Q: Em classification report, compreendi a informação da coluna 'precision'. Mas o que as outras trazem de informação? Tanto as colunas restantes como as linhas abaixo de B e M, não consegui compreender essa tabela muito bem. E sobre a confusion matrix, o que ela me informa exatamente?

R: A coluna que contém B e M é a coluna que indica as suas classes preditas, no caso do problema do diagnóstico é (B)enigno e (M)aligno. Para cada uma dessas classes é calculado o precision recall e F-Measure considerando a classe alvo. Pois essas métricas podem apresentar resultados diferentes dependendo qual classe é a alvo. Pensando em um problema de classificação de SPAM, onde queremos classificar se um e-mail é SPAM ou não. A variável alvo seria SPAM, com duas classes Sim e Não, e como queremos identificar e-mail SPAM a classe positiva é sim. A precisão então indicará o quanto seu modelo conseguiu acertar dentre todas classificações positivas que ele fez, ou seja, a precisão é uma boa medida para determinar, quando os custos do Falso Positivo são altos. Por exemplo, detecção de spam de e-mail. Na detecção de spam de e-mail, um falso positivo significa que um e-mail que não é spam (negativo real) foi identificado como spam (spam previsto). O usuário de email pode perder emails importantes se a precisão não for alta para o modelo de detecção de spam. Quando tratamos de RECALL, estamos verificando como o seu modelo está se comportando no momento que ele erra a classificação positiva. Recall será a métrica do modelo que usamos para selecionar nosso melhor modelo quando há um alto custo associado a Falso Negativo. O F1 Score é necessário quando você deseja buscar um equilíbrio entre Precision e Recall.

** FAQ gerado com base em comentários até o dia 30/11/2022.*