Faculdade de
Medicina da USP

Instituto do
Câncer do Estado de São
Paulo

inteli
instituto
de tecnologia
e liderança

Controle do Documento

Histórico de revisões

Data	Autor	Versão	Resumo da atividade
11/08/2022	Luísa Vitória Leite Silva, Moisés Caze e Bianca Casemiro Lima.	1.0	Objetivos, Justificativa e preenchimento da seção 4
28/08/2022	Luísa Vitória Leite Silva e Moisés Caze.	4.3	Preparação dos dados
13/09/2022	Allan dos Santos Casado, Bianca Cassemiro Lima e Felipe Silberberg.	4.4 e 4.5	Modelagem e avaliação dos resultados
28/09/2022	Allan dos Santos Casado, Felipe Silberberg, Gabriel Rocha Pinto Santos e Moisés Caze	4.4 e 4.5	Hiper parametrização e comparação da modelagem e avaliação dos resultados
05/06/2022	Bianca Casemiro Lima e Luísa Vitória Leite Silva	3, 5 e 4.6	Metodologia, Conclusão e Comparação de modelos

Sumário

1. Introdução

Nosso parceiro de mercado é a Faculdade de medicina da USP (FMUSP), que foi fundada em 1912 e o prédio da faculdade fica em Pinheiros. A instituição possui 1.000 funcionários sendo 368 docentes, 1.400 alunos de graduação, mais de 2.000 alunos de pós-graduação e mais de 1.600 residentes.

De forma resumida, segundo o site da FMUSP, sua visão é formar profissionais da saúde com uma forte formação e com muita ética e humanismo. Em relação à missão, eles têm como objetivo um ensino de graduação e pós-graduação. Dessa forma, o ensino é relacionado à medicina, fisioterapia, fonoaudiologia e terapia ocupacional, com foco voltado para pesquisa, cultura e extensão de serviços à comunidade. Seus pilares são: ética, respeito ao indivíduo, humanização, honestidade, pioneirismo e excelência, respeitando os mais elevados preceitos éticos e morais.

O problema proposto pelo parceiro se relaciona ao câncer de mama e, por meio de um modelo preditivo, devemos apresentar futuros prognóstico da paciente que está passando pelo tratamento.

2. Objetivos e Justificativa

2.1. Objetivos

Os principais objetivos do parceiro de negócio consistem em fazer uma análise dos dados da paciente de câncer de mama e, a partir dessa análise, conseguir receber uma resposta sobre a taxa do tempo de sobrevida. Além disso, um direcionamento sobre o agendamento de consultas futuras será apresentado, levando em conta o status do paciente e os possíveis desdobramentos do tratamento.

2.2. Justificativa

Levando em conta os objetivos citados no item anterior, a nossa solução proposta é a criação de um modelo preditivo que consiga analisar os dados da paciente e, a partir disso, fazer uma análise preditiva sobre a taxa do tempo de sobrevida. Não obstante, com base nos dados principais e estruturados, o modelo conseguirá direcionar o médico sobre os possíveis agendamentos de consulta.

3. Metodologia

3.1. CRISP-DM

A metodologia CRISP-DM está relacionada com a mineração de dados. Ela soluciona problemas da área de ciências de dados, além de possuir formato cíclico. Sua principal característica é a flexibilidade, pois depois de terminado um ciclo, caso os objetivos não sejam atingidos, o ciclo pode ocorrer novamente. Este método possui seis etapas:

1- Business Understanding (entendimento do negócio): Essa etapa tem uma visão 100% estratégica, ela foca em entender o negócio, seus objetivos e seus problemas. Ela tem uma grande importância pois aqui definimos o objetivo final.

2- Data Understanding (Entendimento dos Dados): Nessa parte nos familiarizamos, entendemos e avaliamos a qualidade dos dados. Assim, hipóteses são criadas e insights são formulados sobre os dados.

3- Data Preparai-o (Preparação dos Dados): Aqui é feita uma seleção, integração, limpeza e transformação nos dados, o que inclui: buscar anomalias nos dados, normalizar, excluir colunas com muitos nulos e agregar os dados.

4- Modeling (Modelagem): Nessa fase, aplicamos várias técnicas de modelos preditivos (a preparação de dados tem uma grande influência nessa quarta fase). O método para aplicar essas técnicas é basicamente o seguinte: selecionar um método -> separar um conjunto de dados para teste -> construir e treinar o modelo.

5- Evaluation (Avaliação): Nessa etapa, deve ser feita a avaliação dos modelos treinados. Essa avaliação é feita por analisando as métricas de desempenho do modelo como: acurácia, precisão, sensibilidade, AUC e outras. A partir dessa análise, é possível visualizar onde o modelo mais erra e isso permite sua alteração, a fim de obter um melhor desempenho.

6- Deployment: Por fim, após todos os ajustes do projeto, é preciso colocá-lo em produção para que os usuários possam utilizá-lo no dia a dia.

3.2. Ferramentas

A principal ferramenta utilizada é o Google Colaboratory, um serviço de nuvem gratuito hospedado pelo próprio Google, com enfoque no desenvolvimento de machine learning e inteligência artificial. É uma ferramenta que permite a mistura de código fonte e texto, geralmente em markdown, com imagens e links. Esse serviço é gratuito e permite que qualquer pessoa com acesso à internet consiga escrever códigos python, importar as bibliotecas e módulos dessa linguagem. Além disso, é possível documentar o que foi feito concomitantemente e de maneira profissional, com textos e imagens.

Além do Google Colaboratory, o GitHub é essencial. Esse é um serviço baseado em nuvem que permite que os desenvolvedores colaborem e façam mudanças em projetos compartilhados enquanto mantêm um registro detalhado do seu progresso. Basicamente, é um repositório compartilhado que permite que múltiplos usuários façam alterações dos códigos, documentos e arquivos relacionados ao projeto.

3.3. Principais técnicas empregadas

A primeira técnica empregada é a feature engineering, em que preenchemos os dados nulos e criamos um arquivo com separação das colunas categóricas e identificadoras em data, alvo e número. Também removemos colunas com uma grande quantidade de nulos, pois seriam colunas divergentes para o modelo. Finalmente, para preenchermos o resto de variáveis nulas, aplicamos a técnica de preencher os dados com base em alguns tipos de normalização: pelo último registro na base de dados, pelo maior valor, pela mediana ou pela média dos valores.

Os quatro principais algoritmos de classificação que utilizamos foram: KNN (K-Nearest Neighbors), Árvore de Decisão, Random Forest (Floresta Aleatória) e SVM (Support Vector Machines). Além disso, as métricas de avaliação aplicadas foram: Precisão, Revocação (Recall), F1-Score, Acurácia e AUC (Curva ROC). No entanto, por ser praticamente inviável levar em consideração as cinco métricas, nós adotamos a acurácia como a única a ser analisada e comparada pelo fato de ser a que mais se adequa ao nosso problema, de acordo com a nossa visão.

4. Desenvolvimento e Resultados

4.1. Compreensão do Problema

4.1.1. Contexto da indústria

Após uma análise de mercado, não identificamos concorrentes diretos à Instituição de Medicina USP, porém pode-se levar em conta como possíveis opositores médicos e instituições que não praticam medicina de maneira ética utilizando tratamentos alternativos. Além disso, diversas Instituições particulares utilizam de novas tecnologias para auxiliar em suas consultas e isso está sendo implementado aos poucos em hospitais públicos por questões de verba.

Concomitantemente, há diversos hospitais que realizam serviços parecidos com o cliente em questão como:

- Hospital São Camilo Oncologia - Unida
- Hospital A.C. Camargo Câncer Center
- Instituto de Câncer de São Paulo

a) A ameaça de novos entrantes é baixa, pois se trata de uma instituição pública, portanto novos hospitais não seriam uma ameaça ao Hospital da USP. Entretanto, quando falamos de hospitais particulares a história é diferente, pois estes competem entre si pelos melhores números para dessa maneira receber reconhecimento.

b) A ameaça de novos produtos é baixa. Levando em conta que se trata de um hospital, os serviços realizados por essa instituição são considerados de extrema importância e um direito básico para a sociedade segundo o Art.196 da nossa Constituição Federal. Portanto, a ameaça de entrada de novos produtos é baixa.

c) Poder de barganha dos compradores, em razão do hospital atender uma grande parcela da sociedade consideramos esse público pulverizado, por esse motivo há um baixo poder de barganha dos compradores.

d) Poder de barganha dos fornecedores - o poder de barganha dos fornecedores é alto, pois as instituições responsáveis por fornecer as máquinas do hospital tem um alto poder nesse sentido. Porém, isso não acontece quando falamos da indústria farmacêutica que tem um poder mais baixo por ter uma concorrência maior.

e) Rivalidade entre competidores - o preço dos serviços oferecidos de certa forma é tabelado e é custeado pelo governo, não só isso, mas como é um serviço público tal força diminui ainda mais, portanto a rivalidade entre competidores é baixa. Levando em conta a excelência do hospital da USP percebemos que ele está presente entre os 10 melhores hospitais do Brasil. Segundo os próprios médicos do hospital da USP, os principais competidores são médicos que utilizam medicina alternativa, além das instituições que não praticam a medicina corretamente.

As principais tendências na área da saúde são:

- Uso da robótica para auxiliar nas cirurgias

- Uso de IA para identificar alvos potenciais de determinadas doenças, o aprimoramento de tratamentos e, principalmente, na detecção de diagnósticos.
- Atendimento médico remoto

4.1.2. Análise SWOT

4.1.3. Planejamento Geral da Solução

Uma pesquisa da CNN Brasil revelou que, segundo a diretoria da OMS, são quase 600 mil pessoas todos os anos que desenvolvem quadros de câncer e, dentre eles, o câncer de mama está entre os mais recorrentes. A Faculdade de Medicina da USP nos apresentou dados de prontuários médicos de pacientes que têm ou tiveram câncer de mama e utilizaram a rede pública de saúde desde 2008. Os *stakeholders* estão em busca de resultados preditivos, a partir da análise dos dados, sobre o tempo de sobrevida estimado em casos individuais.

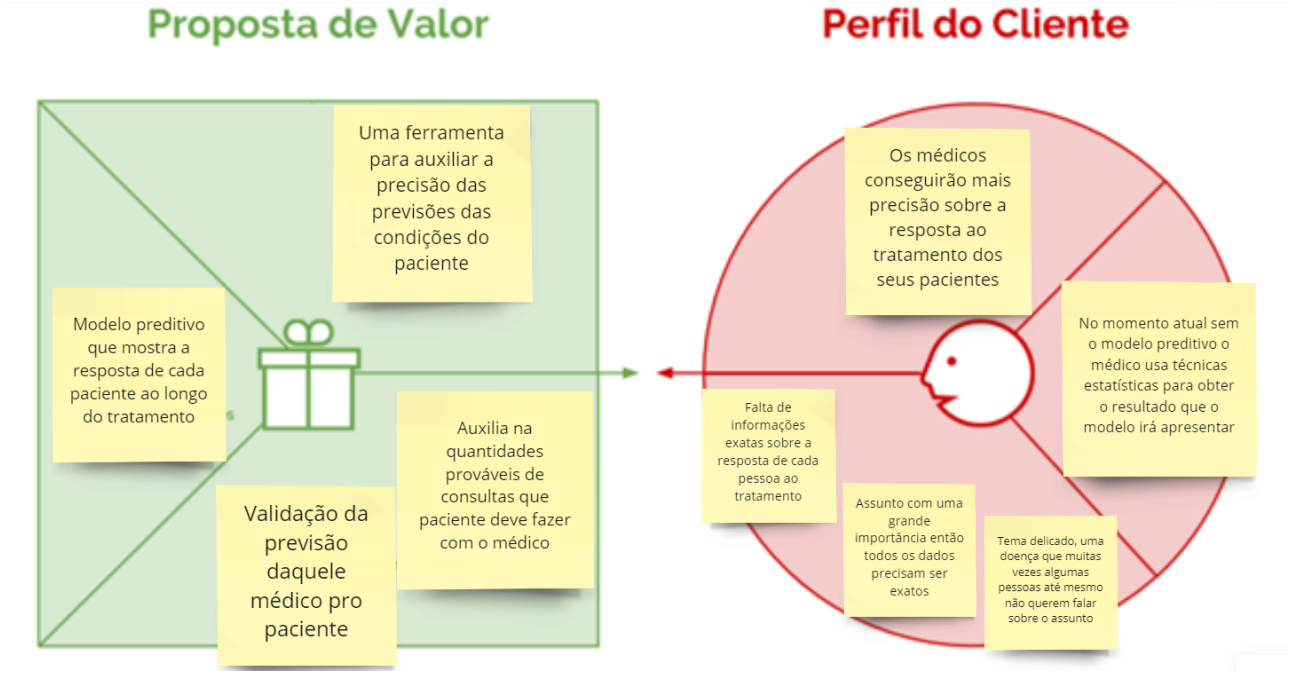
Baseada em mais de 60 mil linhas de registros de cerca de 3 mil pacientes diagnosticados, nossa solução usa padrões identificados para estimar tempo de sobrevida, variabilidade da evolução do câncer de mama, da sua resposta aos tratamentos convencionais e marcadores que possam impactar a qualidade de vida, bem

como a possibilidade de deixar sequelas. Nossa solução criará um score para classificar o tempo de sobrevida como alto ou baixo.

Por parte do Inteli, será desenvolvida uma inteligência artificial que classifica a sobrevida. Com isso, o trabalho dos médicos será facilitado, pois a estimativa de tempo de sobrevida dá ao médico assistência para administrar o tempo entre as consultas para acompanhamento. Com isso, aos stakeholders, médicos e pacientes, serão garantidos eficiência, eficácia e alcance na descoberta e luta contra a doença.

A inteligência artificial terá como critério de sucesso uma avaliação de, pelo menos, 0,8 na curva ROC. Desta forma, tornando claro o intervalo entre as consultas, o tempo será otimizado, trazendo maior número e qualidade para as consultas.

4.1.4. Value Proposition Canvas



4.1.5. Matriz de Riscos

Probabilidade	Ameaças					Oportunidades				
90%	Média	Média	Alta	Alta	Alta	Experiência em trabalhar com uma instituição renomada	Baixa	Baixa	Média	Média
70%	Baixa	Média	Média	Baixo conhecimento técnico das ferramentas	Modelo ter um baixo grau de confiança	Baixa	Experiência de trabalho em equipe	Média	Média	Alta
50%	Baixa	Baixa	Ter que coletar novos dados devido a baixa qualidade dos atuais	Alta	Faltou tempo para desenvolver	Baixa	Baixa	Identificar padrões nos dados para gerar insights relevantes	Alta	Alta
30%	Baixa	Baixa	Média	Pouco comprometimento da Medicina USP com a equipe	A equipe não ter bem definido quais são as métricas de avaliação do modelo	Conseguir novas variáveis que sejam mais relevantes para o treino do modelo	Conseguirmos prever outras variáveis além das esperadas	Média	Alta	Alta
10%	Baixa	Baixa	Falta de comprometimento do grupo em entregar as tarefas	Baixa	Ocorrer vazamento dos dados disponibilizados	Expansão do projeto para pacientes com outros tipos de câncer	Alta	Alta	Alta	Alta
	Muito Baixo	Baixo	Moderado	Alto	Muito Alto	Muito Alto	Alto	Moderado	Baixo	Muito Baixo
	Impacto									

4.1.6. Personas

NOME: Alessandro Vieira

IDADE: 45 anos

GÊNERO: Masculino

OCUPAÇÃO: Médico com especialização em Oncologia

"Minha profissão é minha paixão"

Considerações biográficas e comportamentais

- Trabalha com câncer há 15 anos
- Fluente em 3 idiomas
- Formado em Medicina pela USP em 1988

Dores/Motivações atuais com o problema:

- Quer proporcionar mais confiança aos pacientes
- Não tem uma ferramenta para validar sua provisão para o resultado do pro diagnóstico do paciente
- Ter uma base para saber quantas consultas deve fazer com cada paciente

Objetivos/necessidades específicas em relação ao problema:

- Otimizar o tempo de programar quantas consultas o paciente irá ter
- Consultas realmente necessárias com o paciente
- Atender mais pacientes devido ao tempo otimizado

NOME: Catarina Torres

IDADE: 40

GÊNERO: Feminino

OCUPAÇÃO: Mãe e professora

"Catarina é mãe de dois filhos e recentemente foi diagnosticada com câncer de mama."

Considerações biográficas e comportamentais

- Fumante
- Prática pouco exercício físico
- Muito preocupada
- Passou da menopausa
- Desatenta consigo mesma

Dores/Motivações atuais com o problema:

- Deseja prolongar o seu tempo de vida
- Vive cansada e/ou debilitada
- Evitar idas desnecessárias ao médico

Objetivos/necessidades específicas em relação ao problema:

- Predições ao resultado do tratamento mais confiável
- Prolongar seu tempo restante de vida

4.1.7. Jornadas do Usuário

Alessandro Vieira

Cenário: o médico oncologista irá analisar os exames feitos pela paciente, após isso utilizará a solução de análise preditiva para tornar claro o intervalo entre as consultas.

Expectativas

Expectativas: o médico utiliza a ferramenta e recebe com facilidade o resultado mostrando a subdivisão de risco da paciente e sua taxa de sobrevida.

FASE 1 (Obtenção de dados)	FASE 2 (Análise dos dados referentes aos exames)	FASE 3 (Definição do diagnóstico na visão do médico)	FASE 4 (Consulta da solução de análise preditiva)	FASE 5 (Comunicação do diagnóstico final)
1. Definir quais são as variáveis que serão analisadas 2. Consultar os exames do paciente buscando pelos dados definidos anteriormente 3. Organizar e estruturar os dados	1. Analisar os exames feitos pelo paciente 2. Procurar por padrões nos dados 3. Analisar diagnósticos anteriores	1. Entender todos o sintomas do paciente 2. Conforme a sua experiência define qual o status da paciente 3. Valida o diagnóstico com outros médicos envolvidos nos exames da paciente	1. Utilizar a solução de análise preditiva preenchendo os dados solicitados 2. Receber o resultado do modelo preditivo sobre a condição da paciente e sua taxa de sobrevida 3. Analisar seu diagnóstico em conjunto com o diagnóstico gerado pela solução	1. Dizer o diagnóstico final à paciente 2. Mostrar a solução para a paciente como forma de validação
Essa etapa é muito desgastante mas é de extrema importância.	Essa etapa é desafiadora mas é importantíssima para o diagnóstico.	Nessa fase são muitos os fatores que interferem no diagnóstico por isso atenção é imprescindível	Fico muito feliz com essa nova solução que chegou ao meu escritório, agora posso validar meu diagnóstico!!!	Essa é sempre a fase mais difícil, dizer o diagnóstico sempre me deixa aflito

Oportunidades

- Trazer uma nova tecnologia para a área da saúde.
- Fazer com que os médicos tenham um novo meio de consulta.
- Fazer com haja mais veracidade no diagnóstico e no tempo de sobrevida da paciente.

Responsabilidades

Cabe ao time desenvolvedor da solução garantir a assertividade do modelo preditivo por meio de um bom treinamento da máquina e em casos de ambiguidade no resultado trazer o resultado que mais fará sentido à paciente.

4.2. Compreensão dos Dados

1. O arquivo contendo a base de dados foi disponibilizado no formato xlsx. Para a leitura no pandas, o formato do arquivo foi transformado para csv (comma separated values). Estes dados são provenientes do banco de dados do Instituto do Câncer de São Paulo/Faculdade de Medicina da USP, com informações provenientes de pacientes mulheres diagnosticadas com câncer de mama, contendo 104 colunas e 61683 registros que totalizam 3769 pacientes únicos, dado que um paciente possui múltiplos registros. As colunas se referem a variáveis como subtipo do câncer, escolaridade, histórico de gravidez, histórico de consumo de álcool e tabaco, entre outros.

1.1. Foram disponibilizadas três bases de dados: uma delas é a descrita anteriormente, que possui dados sobre as pacientes e sobre seu quadro de saúde; a segunda, por sua vez, possui informações sobre altura, peso e a data em que essa coleta foi feita. Por fim, a terceira possui as datas das consultas das pacientes. Todas essas tabelas possuem o atributo "record_id", identificador das pacientes, possibilitando a mesclagem das tabelas por meio desta chave primária.

1.2. Em relação aos dados dispostos, é possível observar grande presença de dados nulos na maioria das colunas. Isso impossibilita algumas análises e torna o processamento dos dados mais trabalhoso devido a necessidade de lidar com esses valores. Além disso, gera o risco de não possuirmos uma quantidade de dados significativa para o treino do modelo preditivo ou de um treino enviesado, devido a um tratamento inadequado desses valores.

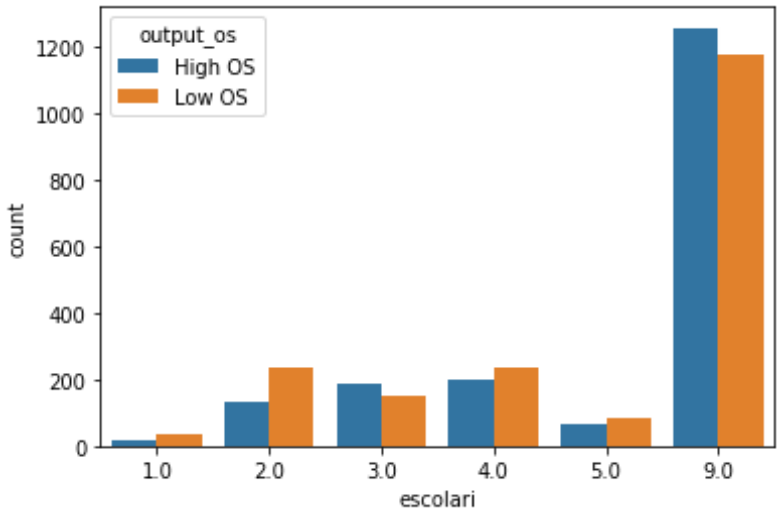
Os dados coletados representam pacientes que acessam a rede pública de saúde. Por termos muitas variáveis, um risco pode ser a utilização de algumas irrelevantes para o treinamento do modelo.

1.3. Em relação à quantidade de registros, iremos fazer as análises baseadas em toda a base de dados, pois existem 3769 pacientes, o que não é uma quantidade tão grande que impossibilite a utilização do conjunto completo. Além disso, como possuímos muitas variáveis, decidimos inicialmente filtrá-las com base nos atributos que os especialistas da área disseram que são mais relevantes para o problema que está sendo abordado. Algumas delas são: subtipo e estádio do câncer, histórico de gravidez e menstruação e consumo de álcool e tabaco.

1.4. Para garantir a confidencialidade das pacientes, os dados foram anonimizados. Ademais, os dados disponibilizados são confidenciais e para uso único e exclusivo no desenvolvimento do projeto. Logo, não podem ser divulgados para o público.

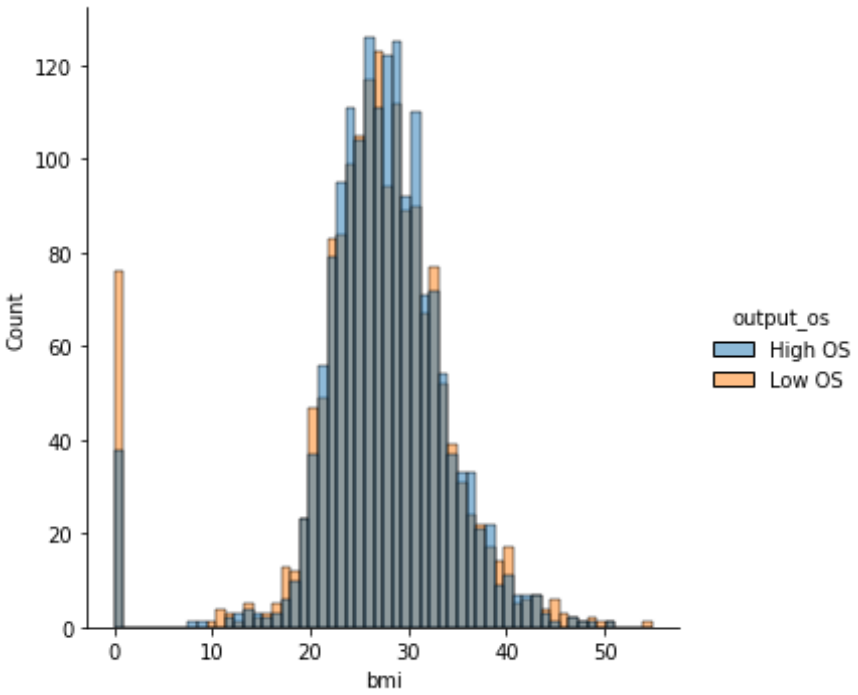
2. Nossos processos da feature engineering se deram, principalmente, pelo preenchimento de dados nulos das pacientes baseado em algum outro registro em que eles estivessem declarados. Em caso de algum dado que se manteve fixo, como a escolaridade, ou em caso de dados contínuos variáveis, como o índice de massa corporal (imc ou bmi), em que calculamos a mediana entre os valores e usamos o resultado para calcular quando necessário. No processo de análises de correlações, buscamos identificar algumas tendências dentre as variáveis, então testamos algumas:

- A relação de **grau de escolaridade** das pacientes varia de acordo com seu **tempo de sobrevida**, pois imaginamos que o nível de escolaridade pode, por vezes, afetar a qualidade de vida da paciente. Segue o gráfico:



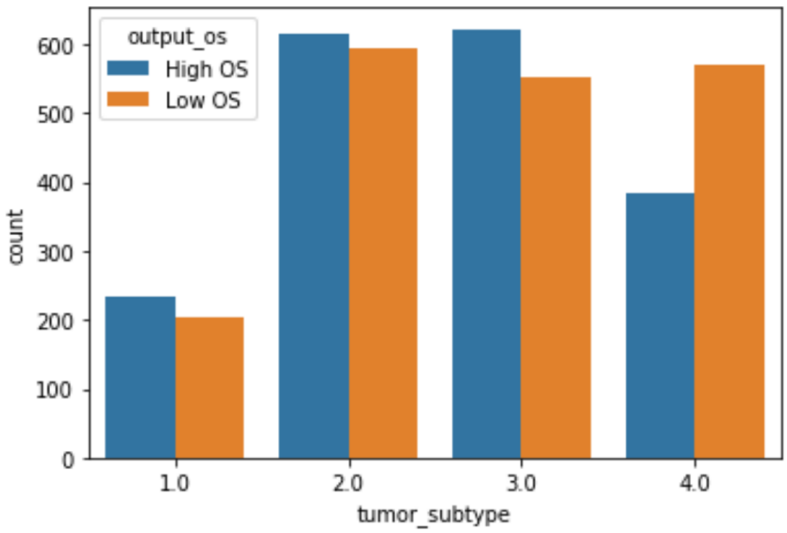
Relação da taxa de sobrevida em função da escolaridade

- A relação entre o **índice de massa corporal** (imc ou bmi) com a **taxa de sobrevida**, visando verificar se os extremos na escala do imc afetam ou não o tempo de vida da paciente. Segue o gráfico:



Relação da taxa de sobrevida em função do IMC (índice de massa corporal).

- Proporção para cada **subtipo** de acordo com o **tempo de sobrevida**, buscando identificar a proporção de tempo de sobrevida por subtipo do câncer(gravidade). Segue o gráfico:

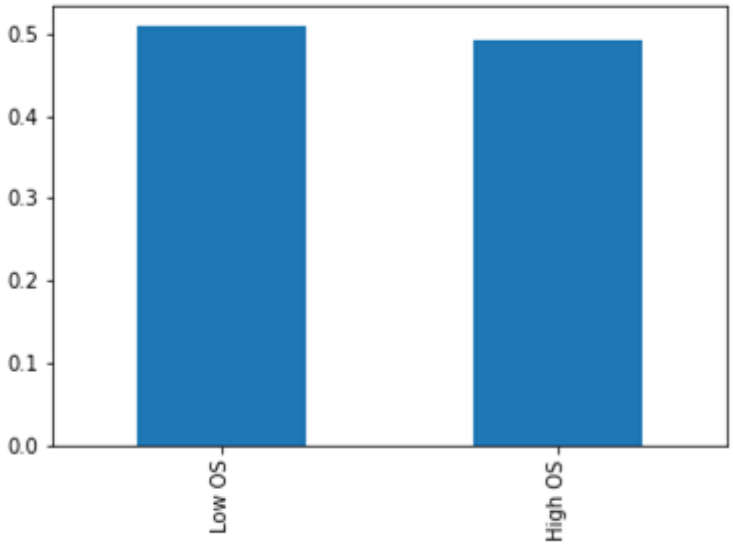


Relação do tempo de sobrevida em função do subtipo do tumor

3. De acordo com o Instituto nacional do câncer, o tempo de sobrevida (overall survival) é definido como:

   "The length of time from either the date of diagnosis or the start of treatment for a disease, such as cancer, that patients diagnosed with the disease are still alive. In a clinical trial, measuring the overall survival is one way to see how well a new treatment works. Also called OS".
   "O tempo decorrido entre o diagnóstico e o tratamento da doença, o câncer nesse caso, é determinante para a sobrevida do paciente. Num ensaio clínico, medir a taxa de sobrevida é um meio para ver como um novo tratamento funciona. Também chamado de OS". Em versão livre para o português.

- A variável target é "output_os";
- Ela possui dois valores possíveis (High OS e Low OS), o que confere uma natureza binária e faz com que tenhamos uma tarefa de classificação;
- Não possuímos registros em que ela é nula;
- Essa variável é bem balanceada, pois está em uma proporção quase de 50/50;
- Gráfico da proporção de possíveis valores da variável target abaixo:



Relação de possíveis valores para a variável target.

4.3. Preparação dos Dados

Movimentação dos Dados

https://colab.research.google.com/drive/1L5BF84fwp4jhwsOLHb0FIAXdmRflk3xx#scrollTo=kUAtA6yhJGif

O primeiro notebook usado na sequência é o “Movimentação dos Dados”. Esse notebook consiste em copiar a base de dados, que naquele instante se encontra no drive da turma, e escrevê-la em um novo arquivo no repositório usado pelo grupo para desenvolvimento. Como o próprio sugere, faz a movimentação da base de dados e se resume a dois elementos:

Uma célula que lê a base de dados disposta no drive da turma e a salva em uma variável:

```
[7]  data = pd.read_csv('/content/drive/MyDrive/Turma 2022.1 - Preditivo - USP/Base/NEW-BDIPMamaV11-INTELI.csv')
```

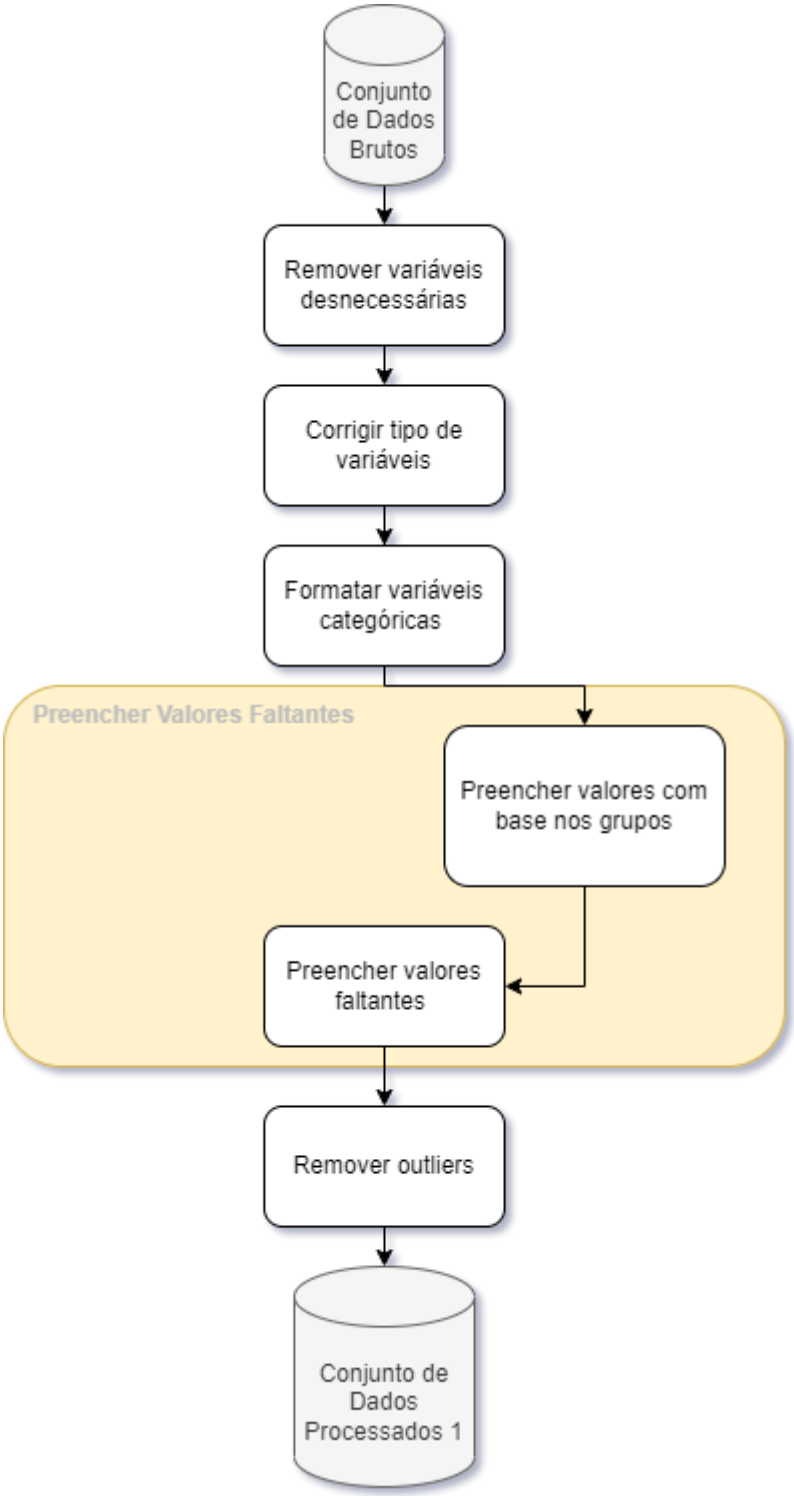
E, em seguida, escreve o arquivo no repositório de desenvolvimento do grupo:

```
[14] data.to_csv('/content/drive/MyDrive/MedicinIA/Sample_Data/Raw/DataSet0.csv',
                 index=False)
```

Por último, nesse mesmo notebook criamos a variável target (output_os) e, por fim, geramos um novo csv, desta vez com a variável target.

Limpeza de Dados

https://colab.research.google.com/drive/1ePOBQsJ14R6nGeKnctlPLnLj5QP7IELo



Nesta segunda etapa, os dados brutos do repositório do grupo foram carregados e foi criado um arquivo, o ". yaml", um tipo de arquivo usado para a serialização de dados. Nesse, as colunas foram discriminadas por tipo (categóricas, numéricas, identificadoras, alvo e colunas com data). Segue a célula que importou a classificação feita manualmente no arquivo. yaml:

```
data = pd.read_csv('/content/drive/MyDrive/MedicinIA/Sample_Data/Raw/DataSet1.csv')
with open('/content/drive/MyDrive/MedicinIA/Sample_Data/Raw/DataSet1.yaml', 'r') as f:
    config = yaml.safe_load(f)

numerical = config['NumericalColumns']
categorical = config['CategoricalColumns']
identifier = config['IdentifierColumns']
datetime = config['DateTimeColumns']
target = config['Target']
```

A próxima etapa da limpeza de dados baseou-se em remover as colunas com uma quantidade irrelevante de dados não nulos, pois elas seriam inúteis ao nosso modelo. As colunas eliminadas foram as seguintes:

- partos (numérica);
- daughter_n (numérica);
- tempo_repo_hormo_tasy (categórica);
- hormone_therapy_tasy (categoria).

A seguir, a correção de colunas que possuem dados de múltiplos tipos. Uma variável categórica, por não possuir quantidades ou valores que podem ser tratados como números, deve ser tratada com dados do tipo texto. Primeiro, modificamos o tipo das colunas categóricas (antec fam_cancer_mama, rec01, rec02, rec03, rec04, meta01, meta02, meta03 e meta04) e depois removemos suas casas decimais, pois estas fazem com que os dados sejam lidos como números e não textos.

Pela abundância de dados nulos, foram usados vários métodos para que fosse efetuado um preenchimento preciso, alguns deles foram preenchidos a partir do último registro na base de dados, enquanto outros foram pelo maior valor, pela mediana ou pela média dos valores.

Agregação de dados

https://colab.research.google.com/drive/1ooGPU2Y4E0slajSfp-LzBeRYWBiRPEXb

As células iniciais desse notebook se assemelham aos demais, que importam a base de dados e o arquivo de descrição para facilitar a identificação de cada variável pelo seu tipo. Depois disso, foi criado um conjunto para os dados numéricos e outro conjunto para os dados categóricos. Por último, os conjuntos

foram agregados, contendo uma linha para cada paciente, seguindo as métricas a seguir:

Numéricas

```
[ ]  1  agg_numerical = data[numerical+identifier].groupby('record_id').agg(
     2      bmi_median = ('bmi', 'median'),
     3      bmi_std = ('bmi', 'std'),
     4      bmi_min = ('bmi', 'min'),
     5      bmi_max = ('bmi', 'max'),
     6      sister_n = ('sister_n', 'max'),
     7      ki67_perc_mean = ('ki67_perc', 'mean'),
     8      follow_up_days = ('follow_up_days', 'mean'),
     9      follow_up_days_recidive = ('follow_up_days_recidive', 'mean'),
    10      follow_up_days_recidive_std = ('follow_up_days_recidive', 'std'),
    11      height = ('height', 'mean'),
    12      weight_mean = ('weight', 'mean'),
    13      weight_std = ('weight', 'std')
    14  )
```

Segue a tabela resultante com os dados já agregados:

record_id	bmi_median	bmi_std	bmi_min	bmi_max	sister_n	ki67_perc_mean	follow_up_days	follow_up_days_recidive	follow_up_days_recidive_std	height	weight_mean	weight_std
54	29.15	0.435775	27.7	29.8	-1.0	19.6	3585.0	1338.0	0.0	152.100000	67.335000	1.014591
302	24.80	6.650833	22.4	45.0	-1.0	-1.0	2225.0	-1.0	0.0	157.909091	59.916000	2.921812
710	26.40	5.330967	0.0	45.0	-1.0	20.0	3294.0	2442.0	0.0	155.117647	63.451471	1.348497
752	35.90	0.525259	35.1	37.2	-1.0	-1.0	4153.0	-1.0	0.0	152.000000	83.110000	1.198923
1589	22.90	4.715171	22.2	45.0	-1.0	-1.0	3290.0	-1.0	0.0	167.000000	64.173684	1.305334
...	...	...	...	...	...	...	...	...	...	...	...	...
82057	38.80	0.843039	37.4	40.5	-1.0	30.0	441.0	-1.0	0.0	156.000000	94.550000	2.070887
82059	27.55	0.508060	26.8	28.6	-1.0	15.0	351.0	-1.0	0.0	164.000000	74.275000	1.346233
82122	25.25	0.240535	24.7	25.4	-1.0	-1.0	401.0	-1.0	0.0	155.000000	60.400000	0.568624
82205	42.50	15.256890	0.0	45.0	-1.0	-1.0	337.0	-1.0	0.0	174.400000	133.612500	4.425282
82240	31.00	0.000000	31.0	31.0	-1.0	-1.0	425.0	-1.0	0.0	161.000000	80.350000	0.000000

No conjunto das variáveis categóricas, temos 70 colunas, as quais foram distribuídas entre uni e multivaloradas. Para as univaloradas, agrupamos seus valores inserindo em cada linha o valor único respondido para aquele paciente. Nas multivaloradas, por sua vez, agrupamos os múltiplos valores para aquele paciente em um só campo, separando-os por vírgula, como pode ser observado no primeiro registro da coluna tumor_stage, por exemplo. Por serem muitas colunas, apresentamos abaixo apenas algumas delas:

record_id	ultinfo	primary_diganosis	escolari	tobaco	tobaco_type	ultinfo	tumor_stage	tumor_margin	hormone_therapy	morfo	pregnancy_history
54	2	2	2	2	1	2	10,99	1	Não informado	85003,88211	1
302	4	Não informado	2	Não informado	Não informado	4	21	Não informado	Não informado	85003	Não informado
710	2	Não informado	4	Não informado	Não informado	2	31	Não informado	Não informado	85003	Não informado
752	2	Não informado	2	Não informado	Não informado	2	21	Não informado	Não informado	84803	Não informado
1589	2	2	3	Não informado	Não informado	2	22	Não informado	Não informado	85003	Não informado
...	...	...	...	...	...	...	...	...	...	...	...
82057	2	2	Não informado	Não informado	Não informado	2	32	Não informado	0	85003	1
82059	2	2	Não informado	Não informado	Não informado	2	32	Não informado	Não informado	85003	Não informado
82122	2	2	Não informado	Não informado	Não informado	2	21	Não informado	Não informado	85003	Não informado
82205	1	Não informado	Não informado	Não informado	Não informado	1	40	Não informado	Não informado	85003	Não informado
82240	2	2	Não informado	Não informado	Não informado	2	33	Não informado	Não informado	85003	Não informado

134 rows × 11 columns

O mesmo processo foi efetuado para as variáveis que representam data e target, segue a célula que o fez:

```
1  agg_target = data.groupby('record_id')['output_os'].last()
```

```
[ ]  1  datetime_list = list(map(lambda x: list(x.keys())[0], datetime))
     2  agg_datetime = data.groupby('record_id')[datetime_list].last()
```

Por fim, agrupamos todos os conjuntos (categóricas, numéricas, alvo, identificadores e variáveis para data) para que fosse possível a análise de cada paciente com uma linha de registro que resuma todas as demais. Segue uma parte do resultado, que possui 85 colunas tratadas, limpas e agregadas da melhor forma possível, dado o entendimento do negócio no período em que se passava:

record_id	bmi_median	bmi_std	bmi_min	bmi_max	sister_n	ki67_perc_mean	follow_up_days	follow_up_days_recidive	follow_up_days_recidive_std	height	weight_mean	weight_std	abortion	alcohol	alcohol_type___1
54	29.15	0.435775	27.7	29.8	-1.0	19.6	3585.0	1338.0	0.0	152.100000	67.335000	1.014591	Não informado	2	0
302	24.80	6.650833	22.4	45.0	-1.0	-1.0	2225.0	-1.0	0.0	157.909091	59.916000	2.921812	Não informado	Não informado	0
710	26.40	5.330967	0.0	45.0	-1.0	20.0	3294.0	2442.0	0.0	155.117647	63.451471	1.348497	Não informado	Não informado	0
752	35.90	0.525259	35.1	37.2	-1.0	-1.0	4153.0	-1.0	0.0	152.000000	83.110000	1.198923	Não informado	Não informado	0
1589	22.90	4.715171	22.2	45.0	-1.0	-1.0	3290.0	-1.0	0.0	167.000000	64.173684	1.305334	Não informado	Não informado	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
82057	38.80	0.843039	37.4	40.5	-1.0	30.0	441.0	-1.0	0.0	156.000000	94.550000	2.070887	Não informado	Não informado	0
82059	27.55	0.508060	26.8	28.6	-1.0	15.0	351.0	-1.0	0.0	164.000000	74.275000	1.346233	Não informado	Não informado	0
82122	25.25	0.240535	24.7	25.4	-1.0	-1.0	401.0	-1.0	0.0	155.000000	60.400000	0.568624	Não informado	Não informado	0
82205	42.50	15.256890	0.0	45.0	-1.0	-1.0	337.0	-1.0	0.0	174.400000	133.612500	4.425282	Não informado	Não informado	0
82240	31.00	0.000000	31.0	31.0	-1.0	-1.0	425.0	-1.0	0.0	161.000000	80.350000	0.000000	Não informado	Não informado	0

4134 rows × 85 columns

4.4. Modelagem

Inicialmente, escolhemos quatro algoritmos de classificação para realizar uma primeira análise: KNN (K-Nearest Neighbors), Árvore de Decisão, Random Forest (Floresta Aleatória) e SVM (Support Vector Machines). O critério de escolha utilizado está ligado à fácil compreensão e entendimento que estes modelos oferecem, além de bons resultados preliminares.

Após a seleção dos algoritmos para os primeiros experimentos, demos sequência ao processo de modelagem com base no procedimento a seguir:

1. Importação das ferramentas e bibliotecas utilizadas:
    - Pandas (manipulação dos dados)
    - Matplotlib (visualização dos resultados)
    - Scikit Learn (treino dos modelos e avaliação)

2. Importação dos dados:
    - Foi utilizada a base de dados DataSet3, que deriva da tabela principal, manipulada durante a Sprint 2. Segue uma amostra da base:

ki67_perc_last	height	weight	follow_up_days_recidive	follow_up_days_recidive_mean	follow_up_days_recidive_std	sister_n	bmi_mean	bmi_std
15.0	152.100000	67.335000	1338.0	1338.0	0.0	-1.0	29.114000	0.435775
-1.0	157.909091	59.916000	-1.0	-1.0	0.0	-1.0	24.403571	1.296000
20.0	155.117647	63.451471	2442.0	2442.0	0.0	-1.0	25.668421	4.313029
-1.0	152.000000	83.110000	-1.0	-1.0	0.0	-1.0	35.961538	0.525259
-1.0	167.000000	64.173684	-1.0	-1.0	0.0	-1.0	23.040909	0.518844
...	...	...	...	...	...	...	...	...
30.0	156.000000	94.550000	-1.0	-1.0	0.0	-1.0	38.825000	0.843039
15.0	164.000000	74.275000	-1.0	-1.0	0.0	-1.0	27.583333	0.508060
-1.0	155.000000	60.400000	-1.0	-1.0	0.0	-1.0	25.192857	0.240535
-1.0	174.400000	133.612500	-1.0	-1.0	0.0	-1.0	37.900000	15.436598

3. Divisão dos dados:
    - Primeiramente, separamos as colunas “record_id” e “output_os” das demais, sendo essa última a variável target do modelo. Após isso, definimos os dados de treino como sendo todas as colunas restantes da tabela, bem como a proporção de dados de treino/teste (80% dos dados para treino e 20% para teste). Para isso, utilizamos o método train_test_split do sklearn, conforme mostrado abaixo:

```
[ ]  #divisão dos dados entre treino e teste
     x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=3)
```

4. Treino:
    - Após a separação dos dados, importamos e instanciamos os modelos e, por fim, os treinamos utilizando a função .fit() do sklearn.

```python
#instanciando os modelos que serão utilizados para treino
training = {'Decision Tree': DecisionTreeClassifier(random_state=3), 'KNN': KNeighborsClassifier(), 'Random Forest': RandomForestClassifier(random_state=3), 'SVM': SVC(random_state=3)}

#laço que itera sobre cada modelo do dicionário acima
for name, modelo in training.items():
  #treinando o modelo
  modelo.fit(x_train, y_train)

  #fazendo predições com o modelo treinado
  y_pred = modelo.predict(x_test)
```

5. Predições:
    - Com os modelos treinados, utilizamos a função .predict() para realizar previsões em cima dos dados de teste.

6. Avaliação dos modelos:
    - Após as predições, comparamos as predições com os dados de teste da variável target para avaliarmos o desempenho dos modelos.

7. Potencialização dos resultados:
    - Por fim, utilizamos o GridSearch (algoritmo responsável por traçar os melhores hiperparâmetros para cada modelo) para tentar maximizar as métricas, em busca do melhor desempenho possível.

```python
#instancia o algoritmo, passando o modelo e o dicionário com os paramêtros
grid_search = GridSearchCV(estimator = model[0], param_grid = model[1],
                           cv = 2, n_jobs = -1, verbose = 2)
#treinando o algoritmo
grid_search.fit(x_train, y_train)
#selecionando o melhor modelo, com os melhores hiperparâmetros
modelo = grid_search.best_estimator_
```

Decision Tree

Uma árvore de decisão é uma ferramenta de apoio à decisão que usa um modelo de decisões em forma de árvore e suas possíveis consequências, incluindo resultados de eventos aleatórios, custos de recursos e utilidade . É uma maneira de exibir um algoritmo que contém apenas instruções de controle condicional.

Como resultado, obtivemos os seguintes valores:

Acurácia: 71.9 %

Precisão: 72.4%

Recall: 71.5%

F1-score: 71.9%

Auc: 71.9%

Para potencializar os resultados obtidos, modificamos os seguintes hiperparâmetros:

- ‘max_depth’: se refere a profundidade da árvore. Quanto mais profunda uma árvore, mais nós de decisão a árvore possuirá, portanto, existe uma probabilidade maior de ocorrer overfitting. Por outro lado, caso uma árvore não seja tão profunda, existe uma probabilidade maior de ocorrer overfitting, pois existirão poucos nós de decisão e o modelo pode acabar sendo pouco complexo. Para esse hiperparâmetro, testamos os valores 80 e 110.
- ‘max_features’: número de features para considerar ao olhar para o melhor split. Todas as vezes que há uma decisão, o algoritmo busca para um número de features e seleciona a melhor, baseado na impureza de gini ou na entropia. Reduzindo esse número, aumenta-se a estabilidade do modelo e reduz o risco de overfitting. Para esse hiperparâmetro, testamos os valores 2 e 3.
- ‘min_samples_leaf’: o número mínimo de exemplos que cada nó folha deve possuir. Para esse hiperparâmetro, testamos os valores 3 e 5.
- ‘min_samples_split’: o número mínimo de exemplos para fazer o split de um nó. Testamos os valores 8 e 12.

Os melhores hiperparâmetros foram:

- ‘max_depth’ = 80
- ‘max_features’ = 3
- ‘min_samples_leaf’ = 3
- ‘min_samples_split’ = 12

Após o Grid Search:

Acurácia: 73.4 %

Precisão: 73.1%

Recall: 74,8%

F1-score: 73,9%

Auc: 73.4%

Taxa de erro antes e depois do GridSearch:

- Antes: 28.1%;
- Depois: 26.6%.

KNN

O algoritmo de k-vizinhos mais próximos, também conhecido como KNN ou k-NN, é um classificador de aprendizado supervisionado não paramétrico, que usa a proximidade para fazer classificações ou previsões sobre o agrupamento de um ponto de dados individual. Embora possa ser usado para problemas de regressão ou classificação, normalmente é usado como um algoritmo de classificação, partindo do pressuposto de que pontos semelhantes podem ser encontrados próximos uns dos outros. Como resultado, obtivemos os seguintes valores:

Acurácia: 76.3%;

Precisão: 75.5%;

Recall: 78.5%;

F1-score: 77%;

Auc: 76.3%.

Para potencializar esse algoritmo, modificamos os seguintes hiperparâmetros:

- 'n_neighbors': se refere ao número de vizinhos que o algoritmo irá comparar para fazer uma predição. Conforme aumenta-se esse número, as predições do algoritmo se tornam mais estáveis pois haverá mais 'votos'. Testamos os valores entre 1 e 20.
- 'p': qual a métrica de distância que será utilizada. Para p=1 utiliza-se a distância manhattan, para p=2 a distância euclidiana e para qualquer outro valor de p, a distância de minkowski. Testamos os valores 1 e 2.

Os melhores hiperparâmetros foram:

- 'n_neighbors' = 19;
- 'p' = 1.

Após o GridSearch:

Acurácia: 77.1%;

Precisão: 75,4%;

Recall: 80.8%;

F1-score: 78.05%;

Auc: 77.08%.

Taxa de erro antes e depois do GridSearch:

- Antes: 23.7%;
- Depois: 22.9%.

Random Forest

Random Forest ou florestas de decisão aleatória é um método de aprendizado conjunto para classificação , regressão e outras tarefas que opera construindo uma infinidade de árvores de decisão em tempo de treinamento. Para tarefas de classificação, a saída da floresta aleatória é a classe selecionada pela maioria das árvores. Para tarefas de regressão, a previsão média ou média das árvores individuais é retornada. Como resultado, obtivemos os seguintes valores:

Acurácia: 77.4%;

Precisão: 76.4%;

Recall: 79.6%;

F1-score: 78%;

Auc: 77.4%.

Para potencializar esse algoritmo, modificamos os seguintes hiperparâmetros:

- 'n_estimators': se refere ao número de árvores de decisão que serão utilizadas para compor o algoritmo, ou seja, quantos votos serão feitos para realizar a predição.

Os outros hiperparamêtros desse algoritmo são iguais aos da Árvore de Decisão, descritos acima. Os melhores valores foram testados.

Os melhores hiperparâmetros foram:

- 'n_estimators' = 300;
- 'max_depth' = 110;
- 'max_features' = 2;
- 'min_samples_leaf' = 5;
- 'min_samples_split' = 8.

Após o GridSearch:

Acurácia: 77.1%;

Precisão: 75.2%;

Recall: 81.6%;

F1-score: 78.2%;

Auc: 77.1%.

Taxa de erro antes e depois do GridSearch:

- Antes: 22.6%;
- Depois: 22.9%.

SVM

O SVM é um algoritmo que busca uma linha de separação entre duas classes distintas, analisando os dois pontos, um de cada grupo, mais próximos da outra classe. Isto é, o SVM escolhe, entre os dois grupos, o hiperplano que se distancia mais de cada um. Como resultado, obtivemos os seguintes valores:

Acurácia: 76.1%;

Precisão: 72.7%;

Recall: 84.1%;

F1-score: 78%;

Auc: 76%.

Para potencializar os resultados obtidos, modificamos os seguintes hiperparâmetros:

- ‘c’**:** É um hipermetro em SVM para controle de erro. Baixo C indica menos erros e o contrário, mais erros. Isso não significa que um erro baixo traduz um bom modelo, depende totalmente dos conjuntos de dados que o conjunto de dados de erros consiste;
- ‘gamma’**:** É um hiperparâmetro que devemos definir antes do modelo de treinamento que decide a curvatura que queremos em um limite de decisão. Gamma define até onde a influência de um único exemplo de treinamento chega, com valores baixos significando 'longe' e valores altos significando 'próximo';
- ‘kernel’**:** Responsável por transformar os dados de entrada no formato necessário. Alguns dos kernels usados no SVM são lineares, polinomiais e radiais. Para criar um hiperplano não linear, usamos as funções RBF e Polinomial. Para aplicações complexas, deve-se usar kernels mais avançados para separar classes de natureza não linear. Com essa transformação, é possível obter classificadores precisos.

Os melhores hiperparâmetros foram:

- ‘c’ = 1;
- ‘gamma’ = 0.0001.

Após o GridSearch:

Acurácia: 77.1%;

Precisão: 76.8%;

Recall: 78.2%;

F1-score: 77.5%;

Auc: 77.1%.

Taxa de erro antes e depois do GridSearch:

- Antes: 23.9%;
- Depois: 22.9%.

4.5. Avaliação

Em um primeiro momento, decidimos utilizar os modelos Árvore de Decisão, Random Forest, KNN e SVM para uma análise primária. A escolha desses modelos se deve ao fato de serem modelos de classificação de fácil entendimento e interpretação.

Levando em conta a análise de mercado feita na seção 4.1, percebemos que com relação às tendências futuras sobre novas tecnologias, o nosso projeto está totalmente alinhado nesse âmbito. Levando em conta os requisitos propostos sobre como deveria ser o desenvolvimento do projeto, conseguimos cumpri-los trazendo os resultados da taxa de sobrevida que foi descrita em alta ou baixa de acordo com a previsão feita pelo machine learning.

Após as predições feitas, fizemos algumas métricas de avaliação. A partir dessa comparação, chegamos em falsos negativos, falsos positivos, verdadeiros positivos e verdadeiros negativos. Deste modo, conseguimos calcular as seguintes métricas:

1. **Precision (Precisão)**: dentre todas as classificações de classe Positivo que o modelo fez, quantas estão corretas;
2. **Recall (Revocação)**: dentre todas as situações de classe Positivo como valor esperado, quantas estão corretas;
3. **F1-Score**: pode ser descrito como a média harmônica entre precision e recall;
4. **Acurácia**: dentre todas as classificações que o modelo fez, independentemente de classe, quantas estão corretas;
5. **AUC**: área de baixo da curva ROC (Receiver operating characteristic).

4.6 Comparação de Modelos

Levando em consideração a modelagem feita no item 4.4 e as métricas que utilizamos para a análise de dados, fizemos uma tabela comparando os modelos e seus respectivos resultados. Dessa forma, a análise dos dados se torna mais clara e, assim, facilita a determinação de quais modelos se saíram melhores:

	Decision Tree	KNN	Random Forest	SVM
accuracy	0.719581	0.763900	0.774376	0.761483
precision	0.724473	0.755385	0.764977	0.727524
recall	0.715200	0.785600	0.796800	0.841600
f1-score	0.719807	0.770196	0.780564	0.780415
auc	0.719613	0.763742	0.774212	0.760897

Com base nas métricas de avaliação, entende-se que os resultados preliminares foram satisfatórios e atingiram uma porcentagem de acertos maior que a esperada para uma primeira análise. Nesse contexto, o modelo Random Forest se sobressaiu em relação aos demais, visto que obteve as maiores métricas, com exceção do Recall. Isso se deve ao fato de o Random Forest ser um modelo extremamente robusto e se sair bem em grande parte dos casos, uma vez que é do tipo ensemble e combina várias árvores de decisão para obter um resultado mais preciso.

Nesse primeiro momento, definimos a acurácia como a principal métrica a ser considerada para a avaliação do melhor modelo treinado, conforme a orientação passada pelos stakeholders da Medicina USP. Desta forma, por ter obtido 77.4% nesta métrica, optamos pelo modelo que utiliza o Random Forest.

Com uma taxa de erro próxima 0,23, atingimos um resultado muito próximo às expectativas dos stakeholders. Após a adoção dos hiperparâmetros, notou-se uma convergência entre as métricas para os diferentes algoritmos, indicando que a base de dados utilizada está equilibrada e os algoritmos escolhidos são coerentes. Além disso, o GridSearch mostrou-se um bom algoritmo para a seleção de hiperparâmetros, visto que foi capaz de diminuir a taxa de erro em todos os casos.

5. Conclusões e Recomendações

Ao longo do projeto, dedicamos um relevante parcela do tempo à preparação e limpeza da base de dados fornecida, com o intuito de construir um modelo de predição mais fiel e confiável possível. Como consequência, chegamos a um algoritmo de classificação capaz de predizer a sobrevida do paciente com 77.4% de assertividade, com base em features escolhidas de acordo com a proximidade da variável de saída (output_os). Porém, por se tratar de um projeto acadêmico, realizado por alunos recém-graduados do Ensino Médio, é preciso listar algumas recomendações acerca de sua utilização:

- Utilizar o modelo preditivo como um auxiliar da tomada de decisão, mantendo a prioridade ao prognóstico realizado pelo médico;
- Por possuir 77.4% de acurácia, é evidente que o sistema não será capaz de prever todos os casos corretamente.

A fim de garantir uma melhor reação do paciente ao seu diagnóstico, recomenda-se que:

- O médico seja transparente acerca da predição, a fim de garantir uma maior confiança ao paciente;
- O médico esclarece que o modelo auxilia na decisão do melhor tratamento possível.

6. Referências

SERRANO, Layane; ROCHA, Lucas. **Brasil tem quase 600 mil novos casos de câncer por ano, diz diretora da OMS**. CNN Brasil, [*S. l.*], p. 1, 4 fev. 2022. Disponível em: https://www.cnnbrasil.com.br/saude/brasil-tem-quase-600-mil-novos-casos-de-cancer-por-ano-diz-diretora-da-oms/. Acesso em: 11 ago. 2022.

KLUYVER, Thomas; MCKINNEY, Wes. **Pandas: powerful Python data analysis toolkit**. [*S. l.*], 23 jun. 2022. Disponível em: https://pandas.pydata.org/docs/. Acesso em: 12 ago. 2022.

NUMPY COMMUNITY. **NumPy User Guide**. [*S. l.*], 22 jun. 2022. Disponível em: https://numpy.org/doc/1.23/. Acesso em: 12 ago. 2022.

PÁDUA, ANTONIO FRANCISCO LIMA DE OLIVEIRA; SOUSA, FABIANA ARAUJO. METODOLOGIA CRISP-DM: POTENCIALIDADES NA DESCOBERTA DO CONHECIMENTO EM DADOS EDUCACIONAIS. **XVI congresso internacional de tecnologia na educação**, [*S. l.*], p. 1, 19 set. 2018. Disponível em: http://www.pe.senac.br/congresso/anais/2018/pdf/poster/METODOLOGIA%20CRISP-DM%20POTENCIALIDADES%20NA%20DESCOBERTA%20DO%20CONHECIMENTO%20EM%20DADOS%20EDUCACIONAIS.pdf. Acesso em: 5 out. 2022.

SANTOS, Thiago G. Google Colab: o que é, tutorial de como usar e criar códigos. **Alura**, [*S. l.*], p. 1, 22 ago. 2022. Disponível em: https://www.alura.com.br/artigos/google-colab-o-que-e-e-como-usar. Acesso em: 5 out. 2022.

Anexos