**Desafio Binahki**

Parte 1 - Participe da competição Titanic: Aprendizado de máquina com desastres.

Desafio do Titanic é um desafio muito conhecido do Kaggle e é considerado o primeiro desafio para você mergulhar em competições de Machine Learning e se familiarizar como a plataforma kaggle funciona.

Ele traz uma base de dados do famosos acidente marítimo e com ele predizer se a pessoa irá sobreviver ou morrer.

**O problema:**

Usar um modelo de machine learning para tentar predizer com base em algumas features se a pessoa irá sobreviver ou morrer ao acidente.

**Dataset:**

Os dados disponibilizados foram 3 arquivos, 1 de treino com todas as features e também com a feature dizendo se a pessoa sobreviveu ou não, essa usada no treinamento, 1 de teste aqui sem essa feature pois é no kaggle que saberemos o resultado e 1 de exemplo de submissão para o kaggle.

Temos como dados:

- Nome;

- Idade;

- Sexo;

- Onde Embarcou;

- Quanto pagou;

- A cabine;

- Se tinha irmãos/cônjuges a bordo (SibSp - brother, sister, stepbrother, stepsister,

husband, wife);

- se tinha parentes/crianças a bordo (Parch - mother, father, daughter, son,

stepdaughter, stepson);

- A classe;

**Setup:**

- Para todo o processo de análise e treinamento e também a geração do arquivo final foi utilizado o Jupterlab do pacote Anaconda.

- Os testes foram feitos usando um Macbook pro 2017, i5, 8GB de ram, sem o uso de GPU.

**Análise:**

- Analise dos dados:

- O primeiro passo após importar os datasets foi uma análise dos dados usando matplotlib e o seaborn;

- Pude analisar a importância de cada variavel.

- Como cada uma pode ajudar no treinamento;

- Data Cleaning;

- Nessa etapa é onde corrigimos os dados faltantes;

- Media e moda foram as mais usadas;

- Na idade foi usado um agrupamento por Sexo, Classe e Titulo;

- Manipulação dos dados:

Nessa etapa é onde tive o maior trabalho, onde utilizei as referencias para melhorar o score.

- Review 1:

- Algumas variáveis foram modificadas, como por exemplo Age e Fare.

Essas duas dividi por ranges e para cada range atribui um valor, criando assim

uma variável categórica; ([0, 13[ = 0, [13, 25[ = 1 ,....)

- Sex também foi transformada em categórica (female = 1, male = 0)

- Review 2:

- Criação da Variável Title, onde foi pego os títulos de cada pessoa (Mr., Mrs,

Cap, ... ) e foram mapeados para títulos padrões ();

- Variável Family que é somatória de SibSp e Parch;

- Variavel “U” (unknown) em Cabine;

- Por fim uso do get\_dummies;

- Variaveis categóricas em colunas;

**Treinamento:**

Para o treinamento utilizei a maioria dos modelos conhecidos e observei o com o melhor resultados:

- LogisticRegressionCV

- DecisionTreeClassifier

- ExtraTreeClassifier

- RandomForestClassifier

- KNeighborsClassifier

- GaussianNB

- SVC

- CatBoostClassifier

- XGBClassifier

- LGBMClassifier

Resultado:

Model Name Accuracy

LogisticRegressionCV............. 0.8372615039281706

DecisionTreeClassifier........... 0.9090909090909091

ExtraTreeClassifier................ 0.9090909090909091

RandomForestClassifier........ 0.9090909090909091

KNeighborsClassifier............. 0.8484848484848485

GaussianNB........................... 0.7586980920314254

SMV....................................... 0.8338945005611672

CatBoostClassifier.................. 0.8888888888888888

XGBClassifier.......................... 0.9034792368125701

Chart, bar chart

Description automatically generated

Aqui também utilizei informações das referencias:

**Review 1:**

Utilizei 2 modelos diretamente após o treinamento dos com maior acurácia,

RandomForestClassifier e XGBClassifier e obtive como maior score obtive 0.76.

**Review 2:**

Aqui o uso do GridSearch trouxe um ganho no score e também o cross\_validation.

Com o RandomForrest alcancei 0.79186 de score com uma acurácia em treino de

0.8204.