Centro Universitário Carlos Drummond de Andrade

Emerson Da Silva Nascimento

Isabela Maria Ribeiro Da Silva

Leonardo Ferreira Gonzaga

Michael Santos Masanet

Projeto Integrador

São Paulo - SP
2021

Centro Universitário Carlos Drummond de Andrade

Projeto Integrador

Relatório Técnico-Científico apresentado na disciplina de Projeto Integrador para o curso de Tecnologia em análise e desenvolvimento de sistemas do Centro Universitário Carlos Drummond de Andrade (UNIDRUMMOND).

São Paulo - SP
2021

Emerson Da Silva Nascimento, Isabela Maria Ribeiro Da Silva, Leonardo Ferreira Gonzaga, Michael Santos Masanet **Projeto Integrador.** 00f. Relatório Técnico-Científico. Tecnologia em Análise e Desenvolvimento de Sistemas – **Centro Universitário Carlos Drummond de Andrade**. Tutor: Eduardo Palhares. Ponte Rasa, 2021.

RESUMO

O projeto tem como missão avaliar e otimizar uma base de dados com o intuito de analisar a quantidade de pessoas que recebem um salário superior e inferior a $50 mil dólares, utilizando como auxílio uma rede neural onde serão divididas porções diferentes da base de dados, informações de treinamento, validação e predição. Para realizar esse processo será necessário dividir a base de dados em partes como escolaridade, gênero, profissão, etnia, entre outros aspectos e em seguida foi necessário a conversão das respectivas variáveis para valores numéricos para a otimização do da base a remoção de inconsistências presente na mesma. Para que os ajustes sejam possíveis será realizada a demonstração da análise feita pela rede neural. Ao realizar esse processo será utilizada a linguagem de programação Python em conjunto com biblioteca sklearn quem tem como função importar os métodos responsáveis pela rede neural como, por exemplo *KNeighborsClassifier*, *GaussianNB, DecisionTreeClassifier, RandomForestClassifier, LogisticRegression e SVC.*

PALAVRAS-CHAVE: Rede Neural; Python; Base de dados; Validação.

SUMÁRIO

1. INTRODUÇÃO

A estrutura de dados é composta por diversos mecanismos de organização de dados para atender aos diferentes requisitos de processamento. As estruturas são definidas a organização, métodos de acesso e opções de processamento para coleções de itens de informação manipulados pelo programa, isso é utilizado a linguagem de programação Python.

Na linguagem Python podemos utilizar diversos tipos de estrutura de dados, para resolver problemas ocasionais e inúmeras situações durante o processo de desenvolvimento.

No Python as principais estruturas utilizadas são listas que armazena dados em sequência e em cada valor possui um índice, os dicionários que são coleções de itens desordenados que possui uma diferença maior comparada a outro tipo de coleção e tuplas que são estruturas próximas às listas que tem uma característica diferente, onde os elementos inseridos não podem ser alterados.

O problema que será abordado se trata de uma base de dados com valores referentes a um formulário que contém informações pessoais e profissionais como idade, gênero, faixa salarial, ocupação, etc. E com base nessas informações é analisada a quantidade de pessoas que possuem um patrimônio superior a 50 mil dólares, para isso será necessária uma análise de dados e remoção de inconsistências para ser possível obter a informação procurada.

2. DESENVOLVIMENTO

2.1 Objetivos

Desenvolver um algoritmo que possa prever a base salarial de indivíduos que tenham renda superior a 50 mil.

2.2. Justificativa e delimitação do problema

Para o problema abordado tem como extrair informações e eliminar inconsistências na base de dados, visando entender os pontos que influenciam a renda individual dos indivíduos presentes na base de dados.

- formação acadêmica;
- Gênero e etnia.

2. 3. Fundamentação teórica

Como fundamentações teóricas, as aulas ministradas pelo professor Eduardo Palhares, artigos acadêmicos, documentação da linguagem *Python* que nos proporciona uma base de conhecimentos para realizarmos a metodologia e atingirmos os objetivos propostos.

2.4. Aplicação das disciplinas estudadas no Projeto Integrador

No projeto foi possível aplicar o conhecimento e as técnicas ensinadas em aulas e entender de maneira prática como funciona a estrutura de dados e a importância dos algoritmos estudados para atingir os resultados apresentados no projeto.

2.5. Metodologia

Para iniciarmos o projeto foi necessário a utilização da ferramenta Google Colab que permite que qualquer pessoa escreva e execute código Python arbitrário pelo navegador sendo adequado para *Machine learning*, análise de dados e educação.

3. PROJETO

3.1 Inicializando o projeto e importação de base

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
from google.colab import drive
```

código 1 - importação de bibliotecas

A biblioteca *numpy* tem como função realizar operações matemáticas de alto nível, o panda para possibilitar a captura do *dataframe* para tabular linhas e colunas, *matplotlib* que possibilita a visualização de tabelas.

Para ser possível a leitura da base de dados será necessário utilizar a função *read_csv(“”)* que na estrutura de dados é um dicionário do tipo coleção para transformar os valores em binários e trocar todas as ocorrências de seja menor que 50K por 1 e maior por 0.

```
#transforma os salários em valores binarios
income_map = {'<=50K':1,'>50K':0}
df['income'] = df['income'].map(income_map).astype(int)
```

Código - 2: transforma os salários em valores binários

3.2 Análise de variáveis

Para iniciar a análise de variáveis é necessário deixá-las formatadas no padrão *utf-8 para* que não ocorra caracteres especiais nas futuras variáveis.

Para definir as variáveis de gêneros recentes na estrutura foi definido o valor inteiro de 1 para *Male* e 0 para *Female*. Foi necessário aplicar a função *replace()*, para alterar os campos que tinham como inconsistência *"?”* das variáveis *native-country*, *workclass* e *occupation* substituir por NaN e em seguida utilizada *dropna()* para remover as linhas inconstantes da base de dados.

```
#Exibe os registros da tabela
print ("\nTotal registers", df.shape)
```

```
#Substitui as "?" ou informações insuficientes por um NaN
df['native-country'] = df['native-country'].replace('?',np.nan)
df['workclass'] = df['workclass'].replace('?',np.nan)
df['occupation'] = df['occupation'].replace('?',np.nan)
#Remove as linhas com dados insuficientes
df.dropna(how='any', inplace=True)
```

Código - 3: Remoção de linhas com valores inconsistentes

Para a variável *marital-status* foi necessário realizar um *replace()* e substituir os valores *Married-AF-Spouse e Married-Civ-Spouse* por *Couple*. Para a variável Relationship ou realizada *replace()* que enumera os status de relacionamento de 0 a 5 para simplificar a análise.

```
relationship_map={'Unmarried':0,'Wife':1,'Husband':2,'Not-in-family':3,\'Own-child':4,'Other-relative':5}

df['relationship'] = df['relationship'].map(relationship_map)
```

Código - 4: Remuneração dos status de relacionamento da tabela

Para a variável *marital-status* foi necessário realizar um *replace()* e substituir os valores *Married-AF-Spouse e Married-Civ-Spouse* por *Couple*. Para a variável Relationship ou realizada *replace()* que enumera os status de relacionamento de 0 a 5 para simplificar a análise e o mesmo ocorre na variável *Race*.

Para otimizar *workclass* a variável foi necessário utilizar uma função condicional que para analisar melhor as a definição da classe do tipo de organização os indivíduos trabalham como privada ou órgão público.

```
def f(x):
 if x['workclass'] == 'Federal-gov' or x['workclass'] == 'Local-gov' or \
    x['workclass'] == 'State-gov': return 'govt'
 elif x['workclass'] == 'Private': return 'private'
 elif x['workclass'] == 'Self-emp-inc' or x['workclass'] == \
      'Self-emp-not-inc': return 'self_employed'
 else: return 'without_pay'
```

Código - 5: Renomeação do Tipo de organização trabalhista.

A variável capital-gain define e cria imagem que ilustra os ganhos superior e menor que $50.000 dólares por ano.

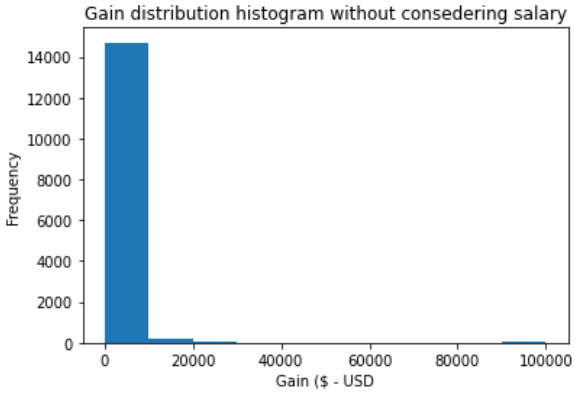

Imagem - 1: Média salarial por ano

A variável capital-loss tem como abordagem realizar distribuição de perdas histograma com redução de salário.

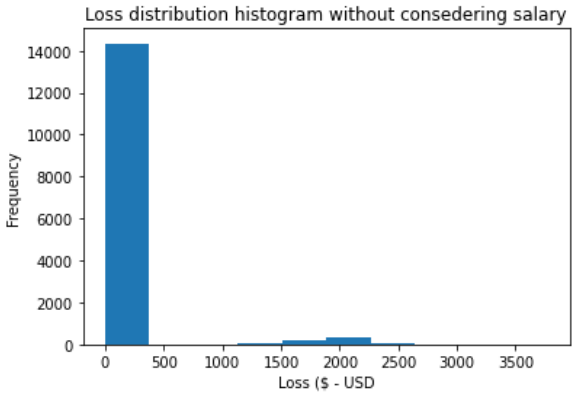

Imagem - 2: Gráfico de redução de salário.

3.3 Removendo variáveis não explicativas

Para iniciar a análise de variáveis é necessário deixá-las formatadas no padrão *utf-8 para* que não ocorra caracteres especiais nas futuras variáveis.

Variável education (nível educacional) e Variável “occupation”(ocupação)

O objetivo dessas variáveis é listar pessoas que não tiveram um longo período de estudo, ou que possuem cargos “básicos”, influenciando diretamente geralmente em uma renda inferior a 50 mil dólares.

Quem possui qualificação nos estudos tem maior possibilidade de atingir a renda maior que 50 mil dólares.

Essas variáveis acabam sendo redundantes, pois, temos outra variável no código que se chama “anos de estudo” que possui basicamente a mesma função, então, iremos realizar a remoção para deixarmos o código mais objetivo.

```
Df.droop(labels=[‘education’,’occupation’],axis=1,inplace=True)
```

Código - 6: Remoção de rótulos especificados das linhas ou colunas..

df.drop = Método para apagar as colunas DataFrame.

O método drop remove/apaga os ‘rótulos’ especificados das linhas ou colunas.

Os ‘rótulos’ podem ser um único rótulo ou rótulos de índice ou de coluna para apagar.

O axis especifica se os rótulos são descartados do índice/linha (0 ou index) ou coluna (1 ou column). index, columns é a alternativa para especificar o eixo. drop(labels, axis=0) é igual a drop(index=labels), enquanto drop(labels, axis=1) é igual a drop(column=labels).
inplace especifica que o DataFrame é modificado no lugar se inplace = True, caso contrário, ele retorna o novo DataFrame com o DataFrame original não modificado.

3.4 Variáveis não discretizáveis

As variáveis “*hours per week*”(horas trabalhadas por semana), “education num" ("número de anos na escola”) e “age” (idade)

O objetivo da variável horas trabalhadas por semana é mostrar a jornada semanal que não exceda 30 horas e a jornada for de até 26 horas semanais ou menos que isso, poderá ser acrescentada de até 6 horas extras complementares semanais.

Na variável número de anos na escola consiste em dizer o número de escolas no estado e no ano de nascimento dos alunos como variável.

A variável idade pode ser avaliada por anos completos e quantitativa (continua), mas se for avaliada através de faixas etárias tipo (0 a 5 anos, 6 1 ou 8 anos) isso tudo vai ser determinado pela maneira que você coletar os dados.

As variáveis descritas acima tem como base este código abaixo:

Ptl.hist(x,bins=nome,destiny=true,histtype=’bar’

Ptl.xlabel, nas variáveis ele descreve as horas trabalhadas o nível da educação, e o nível de idade

Ptl.ylabel descreve a frequência dos funcionários que vão trabalhar e a quantidade de alunos que vão com mais frequência à escola e a idades.

3.5 Analisando a correlação

A análise de correlação dedica-se a inferências estatísticas das medidas de associação linear que seguem coeficiente de relação simples mede a força ou o grau de relacionamento linear entre duas variáveis coeficientes de correlação múltiplo mede a força, ou o grau de relacionamento entre uma variável.

3.6 Construção dos classificadores

Para finalmente conseguirmos uma análise precisa dos dados trabalhados durante todo o projeto, é necessário a utilização de ferramentas presentes na biblioteca *sklearn* biblioteca responsável por construir classificadores para *machine learning*.

Para isso utilizamos uma determinada porcentagem dos dados tratados, onde seriam encaminhados para diferentes atribuições: 50% seriam usados para treinamento, 20% para validação e finalmente 30% seriam utilizados para teste.

```
X = df.drop(['income'],axis=1)
y = df['income']
```

Código - 7: Construção dos classificadores

No código a cima o “X” é atribuído todas as variáveis com exceção da variável *income* , já o “y” é atribuída a variável *income*.

```
#Criação de treino teste do Data Frame
X_train, X_test, y_train, y_test = train_test_split\(X,y,test_size=split_size,random_state=22)
#Criação do treino e validação do Data Frame
X_train, X_val, y_train, y_val = train_test_split\(X_train,y_train,test_size=0.2,random_state=5)
```

Código - 8: Construção dos classificadores

E então separamos a criação de dois *Data Frames*, o primeiro com o objetivo de treino e teste, e o segundo com o intuito de treino e validação, obtendo o resultado de

aproximadamente 25.000 dados no *Data Set* de treino, 6.000 para *Data Set* de validação e por fim 13.000 para o *Data Set* de treino.

Já caminhando para a etapa final do projeto começamos a trabalhar com as bibliotecas de treinamento, mais uma vez iremos importar módulos da biblioteca *sklearn*, mas desta vez iremos utilizar módulos de treino neural, como os: *Nearest Neighbors, Naive Bayes, Decision Tree, Random Forest, Logistic Regression* e *SVC.* E logo em seguida todos esses classificadores são adicionados dentro de uma lista chamada de *models*.

```
models.append((KNeighborsClassifier(n_neighbors=50)))
models.append((GaussianNB()))
models.append((DecisionTreeClassifier()))
models.append((RandomForestClassifier(n_estimators=100)))
models.append((LogisticRegression()))
models.append((SVC()))
```

Código - 9: Construção dos classificadores

Depois desta etapa foi feita a importação das bibliotecas de treinamento e a criação de uma *kfold,* um método de validação cruzada de dados.

E por fim começamos a executar um treinamento onde iremos verificar qual é a margem de acerto no grupo de testes, para seja feita a avaliação dos resultados com o auxílio da matriz de confusão.

A matriz de confusão é uma ferramenta importante para o cálculo de estatísticas como a *accuracy, pressision* e o *recall*, a tabela é dispersa como *tp* sendo *true positve, fp false positive, tn true negative* e *fn false negative.*

Tendo isso em mente foi feita a medição da *accuracy* dos métodos apresentados no código 9 utilizando-se da formula $\frac{tp+tn}{tp+tn+fp+fn}$. Assim como sua precisão $\frac{tp}{tp+fp}$ e revogação $\frac{tp}{tp+fn}$, que irão distinguir os nossos acertos entre os indivíduos que recebem mais ou menos que 50 mil, para que no fim seja feita a média harmônica da precisão e a sensibilidade, também conhecida com f1-score.

4. RESULTADOS

Com o auxílio da matriz de confusão, foi possível medir a porcentagem de certeza dos classificadores utilizados no projeto, evidenciando quais são os métodos mais efetivos para a predição de salário proposta.

O método que apresentou melhor resultado no projeto foi a “floresta aleatória” que previu com cerca de 88% de precisão dos indivíduos que faturam menos de 50 mil e 61% de precisão para os indivíduos que recebem mais de 50 mil, já a rede neural não.

No método SVC obtivemos resultados semelhantes para indivíduos que recebem menos de 50 mil alcançando 86%, porém, não obtivemos um resultado satisfatório para os indivíduos que recebem mais de 50 mil, equivalendo a 0,6%

5. CONSIDERAÇÕES FINAIS

Com base no projeto apresentado podemos observar como o mundo está cada dia mais tecnológico o que pensávamos ser impossível realizar, hoje é possível, e está no nosso cotidiano e são indispensáveis. Esse avanço impacta diretamente na vida da sociedade, trazendo inúmeros benefícios.

No projeto conseguimos identificar como é projetada e implementada uma rede neural para analisar a base de dados proposta que remove inconsistências otimizando o processo de análise para obter resultados mais precisos, com isso reconhecemos a importância da estrutura de dados, e de sua aplicação em análise de estatísticas e diversos projetos.

A linguagem de programação P*ython* foi a ferramenta primordial para realização do projeto porque ela nos permite implementar e otimizar o processo de desenvolvimento de forma prática e objetiva para os fins da realização do projeto gerando resultado proposto em forma de relatório onde o mesmo contempla as informações apresentadas.

Durante o projeto tivemos vários desafios devido inconsistências devido a atualização da biblioteca *sklearn,* que impactou diretamente no trecho *KFold* que tem como objetivo fazer a validação cruzada dos elementos, mas com auxílio do orientador foi possível compreender o problema e realizar a correção do problema.

Com tudo podemos concluir para fins do projeto foi apresentada uma metodologia que teve um resultado satisfatório para o objetivo designado durante o processo de aprendizagem.

REFERÊNCIAS

VASCONCELLOS. PAULO paulovasconcellos.com.br/. 28 comandos úteis de Pandas que talvez você não conheça. *[S.l.]*. paulovasconcellos, São Paulo:2017. Disponível em: https://paulovasconcellos.com.br/28-comandos-%C3%BAteis-de-pandas-que-talvez-voc%C3%AA-n%C3%A3o-conhe%C3%A7a-6ab64beefa93. Acesso em: 21 nov. 2021.

SANTIAGO JR. LUIZ. medium.com.br. Entendendo a biblioteca NumPy. *[S.l.]*. medium, São Paulo:2018. Disponível em: https://medium.com/ensina-ai/entendendo-a-biblioteca-numpy-4858fde63355. Acesso em: 21 nov. 2021.

KRIGER. DANIEL kenzie.com.br. Dicionário Python: o que é, como criar e principais métodos?. São Paulo: kenzie, 2021. Disponível em: https://kenzie.com.br/blog/dicionario-python/. Acesso em: 21 nov. 2021.

GUIZELINI. DIEVAL. bioinfo.ufpr.br/. Matriz de Confusão. São Paulo : bioinfo, 2021. Disponível em: https://www.bioinfo.ufpr.br/moodle/pluginfile.php/725/mod_resource/content/0/05_matriz_de_confusao.pdf. Acesso em: 21 nov. 2021.

LEITE. RODRIGO . drigols.medium.com/. Introdução a Validação-Cruzada: K-Fold. São Paulo: drigols, 2020. Disponível em: https://drigols.medium.com/introdu%C3%A7%C3%A3o-a-valida%C3%A7%C3%A3o-cruzada-k-fold-2a6bced32a90. Acesso em: 21 nov. 2021.