

Treinamento da IA

Os dados que consideramos mais importantes para passar para a IA foram o dia, a hora, a categoria e a distância de uma corrida, esses fatores são os que mais influenciam no preço e portanto são os mais importantes para prever preços de corridas.

Criando o treino e teste

Com a base já limpa e integrada nós criamos o treino e o teste da IA

```
[10] x = df.drop(['Preco', 'ID_Corrída'], axis=1)
     y = df['Preco']

     scaler = StandardScaler()
     scaled_x = scaler.fit_transform(x)

     treino_x, teste_x, treino_y, teste_y = train_test_split(
         scaled_x, y, test_size=0.3, random_state=42)

     display(teste_x)
     display(teste_y)
```

Analisando o desempenho dos modelos

Em seguida usamos esse treino e teste em alguns modelos diferentes para ver qual obteria a melhor performance

```
[16] # Função para fazer a avaliação de um modelo especificado
     def avaliacao(modelo, nome):
         modelo.fit(treino_x, treino_y)

         predicao = modelo.predict(teste_x)
         MAE = mean_absolute_error(teste_y, predicao)
         MSE = mean_squared_error(teste_y, predicao)
         RMSE = np.sqrt(MSE)
         R2 = r2_score(teste_y, predicao)

         print(f'Modelo: {nome}\n')
         print(f'MAE: {MAE:.2f}%')
         print(f'MSE: {MSE:.2f}')
         print(f'RMSE: {RMSE:.2f}')
         print(f'R2: {R2:.2f}')
         print("\n-----\n")

     # Passando diversos modelos pela função de análise de desempenho
     avaliacao(LogisticRegression(), "Regressão Logística")
     avaliacao(KNeighborsRegressor(n_neighbors=5), "KNN")
     avaliacao(DecisionTreeRegressor(), "Árvore de Decisão")
     avaliacao(RandomForestRegressor(n_estimators=100), "Random Forest")]
```

Os modelos testados foram:

- LogisticRegression
- KNeighborsRegressor
- DecisionTreeRegressor
- RandomForestRegressor

Definindo o modelo

O modelo que apresentou a melhor performance foi o RandomForest, então esse vai ser o modelo usado para o projeto

```
[13] # Escolhendo o modelo
      modelo = RandomForestRegressor(n_estimators=100)

      # Treino do modelo escolhido
      modelo.fit(treino_x, treino_y)
```

Modelo: Regressão Logística

MAE: 20.89%
MSE: 15908.85
RMSE: 126.13
R²: -1.78

Modelo: KNN

MAE: 18.08%
MSE: 8271.54
RMSE: 90.95
R²: -0.44

Modelo: Árvore de Decisão

MAE: 16.21%
MSE: 7122.78
RMSE: 84.40
R²: -0.24

Modelo: Random Forest

MAE: 16.05%
MSE: 6655.12
RMSE: 81.58
R²: -0.16

Ainda precisamos refinar mais o treinamento da IA para obter uma melhor performance, mas já temos um framework pronto para um MVP e para poder começar a evoluir

Interface de previsão

Com o modelo já definido e treinado criamos uma pequena interface no próprio Colab para poder entrar com novos dados e fazer previsões novas

```
def interface():
    Categoria = widgets.FloatText(description='Categoria:')
    Dia = widgets.FloatText(description='Dia:')
    Hora = widgets.FloatText(description='Hora:')
    Distancia = widgets.FloatText(description='Distancia:')

    botao = widgets.Button(description='Fazer previsão')

    # Função que define o que acontece quando o botão é clicado
    def click(b):
        entrada = pd.DataFrame([{'ID_Categoria': Categoria.value, 'Dia': Dia.value, 'Hora': Hora.value, 'Distancia': Distancia.value}])

        entrada_scaled = scaler.transform(entrada)
        pred = modelo.predict(entrada_scaled)[0]

        # Verificando qual a categoria de Uber selecionada
        categoria_tipo = ''
        match Categoria.value:
            case 2:
                categoria_tipo = 'UberX'
            case 4:
                categoria_tipo = 'UberBlack'
            case 5:
                categoria_tipo = 'Uber Taxi'
            case 6:
                categoria_tipo = 'Uber Executivo'
            case 8:
                categoria_tipo = 'UberBlack Bag'
            case 9:
                categoria_tipo = 'Uber Comfort'
            case 10:
                categoria_tipo = 'UberFlash'

        print("Resultado\n")
        print(f'Dia: {Dia.value:.0f}')
        print(f'Hora: {Hora.value:.0f}')
        print(f'Categoria: {categoria_tipo}')
        print(f'Dia: {Distancia.value:.0f}')
        print(f'Preço: R${round(pred, 2)}')
        print("\n-----\n")

    botao.on_click(click)
    display(Categoria, Dia, Hora, Distancia, botao)

interface()
```

Interface de previsão de Uber no Google Colab. O formulário permite inserir os seguintes dados:

- Categoria: 4
- Dia: 5
- Hora: 10
- Distancia: 15

Após clicar em "Fazer previsão", o resultado exibido é:

Resultado


Dia: 1
Hora: 10
Categoria: UberX
Dia: 7
Preço: R\$31.68

Resultado

Dia: 5
Hora: 10
Categoria: UberBlack
Dia: 15
Preço: R\$53.9

Link do Colab

O arquivo .ipnyb pode ser encontrado no diretório da entrega 2, segue o link para o notebook

 Treinamento IA Khipo