UDACITY

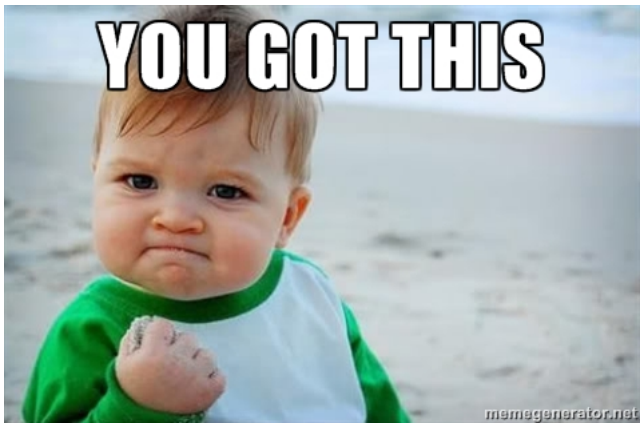
Sair

‹ Return to "Machine Learning Engineer Nanodegree" in the classroom

Finding Donors for CharityML

REVISÃO | HISTORY

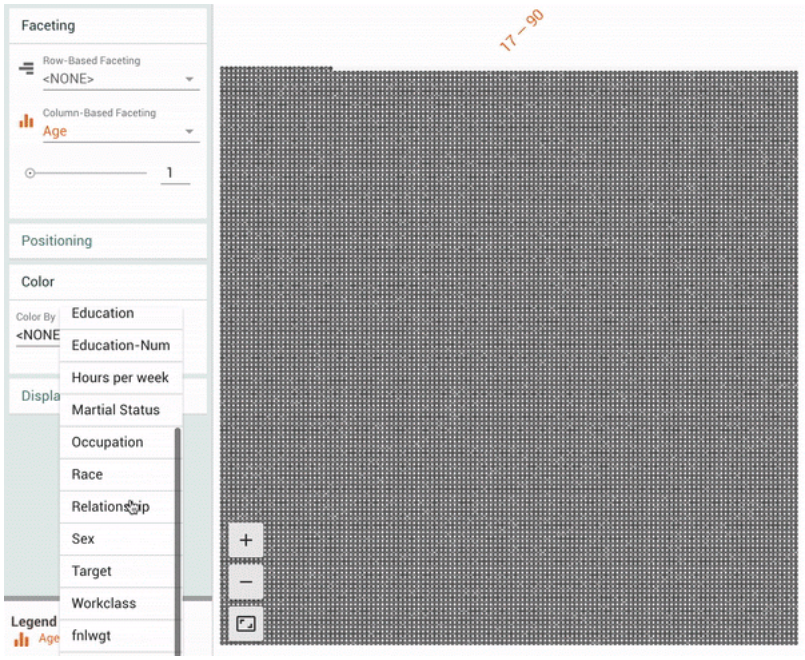
Meets Specifications



Parabéns por passar em seu exame! 👏👏

Bonus

Uma parceria com o Google lançou um software de código aberto para visualizar conjuntos de dados de aprendizado de máquina. E a melhor parte é que eles estão usando o mesmo conjunto de dados deste projeto como live demo! Se você quiser realizar alguma visualização de dados com os dados deste projeto, acesse o link https://pair-code.github.io/facets/:



Exploring the Data

Student's implementation correctly calculates the following:

- **Number of records**
- **Number of individuals with income >$50,000**
- **Number of individuals with income <=$50,000**
- **Percentage of individuals with income > $50,000**

Bônus

Também podemos obter algumas estatísticas iniciais com o método describe do pandas:

```
data.describe()
```

Além disso, uma ferramenta muito legal para visualização de dados e correlações é o pairplot de seaborn (instale usando `pip install seaborn`):

```
In [9]: import seaborn as sns
        sns.pairplot(data, palette='Set1', hue='income')
Out[9]: <seaborn.axisgrid.PairGrid at 0x1169b6050>
```

Preparing the Data

Student correctly implements one-hot encoding for the feature and income data.

Sugestão

Outra alternativa é usar o método `apply` combinado com uma função lambda:

```
income = income_raw.apply(lambda x: 1 if x=='>50K' else 0)
```

Comentário

Esta referência fornece 7 diferentes estratégias de encoding. Binary encoding é uma ótima alternativa para casos em que o número de categorias seja muito alto.

Evaluating Model Performance

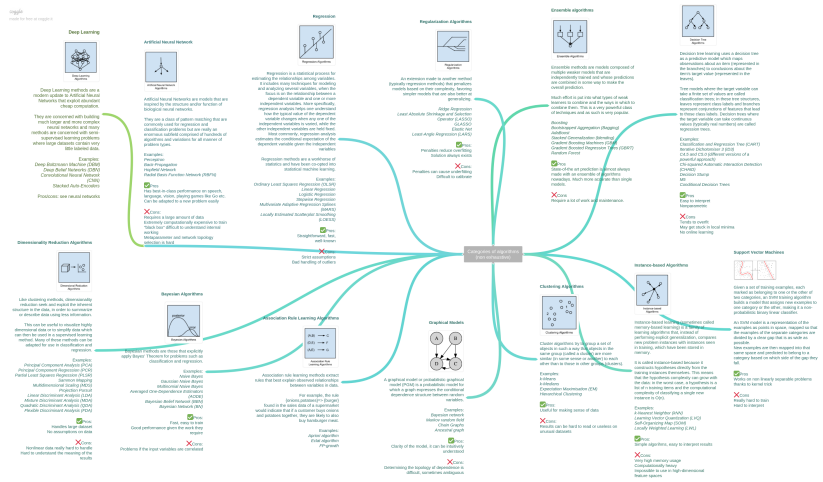
Student correctly calculates the benchmark score of the naive predictor for both accuracy and F1 scores.

The pros and cons or application for each model is provided with reasonable justification why each model was chosen to be explored.

Please list all the references you use while listing out your pros and cons.

Bônus

Você também pode consultar esta referência ao escolher seu estimador:



- Esta referência fornece muitos recursos de visualização de algoritmos de aprendizagem de máquina. O meu favorito é o artigo Gradient Boosting explicado.
- Mário Filho, número 2 do Kaggle no Brasil, fala nesta apresentação sobre os algoritmos favoritos dele.
- Um algoritmo que é bastante popular nas competições do Kaggle é o XGBoost. Este guia é muito bom para bom para calibração deste modelo.
- Este bechmark compara alguns algoritmos de machine learning implementados em R. O algoritmo XGBoost se destaca :)
- Esta referência do sklearn compara as fronteiras de decisões dos principais classificadores.
- Este mapa da sklearn também pode ser útil para dar orientação ao escolher um estimador.

- Três sugestões de livros caso você esteja interessado em mergulhar mais profundamente nas características de cada estimador:
  - The Elements of Statistical Learning
  - Bishop: Pattern Recognition and Machine Learning
  - Machine Learning - Tom Mitchell

Student successfully implements a pipeline in code that will train and predict on the supervised learning algorithm given.

Student correctly implements three supervised learning models and produces a performance visualization.

Improving Results

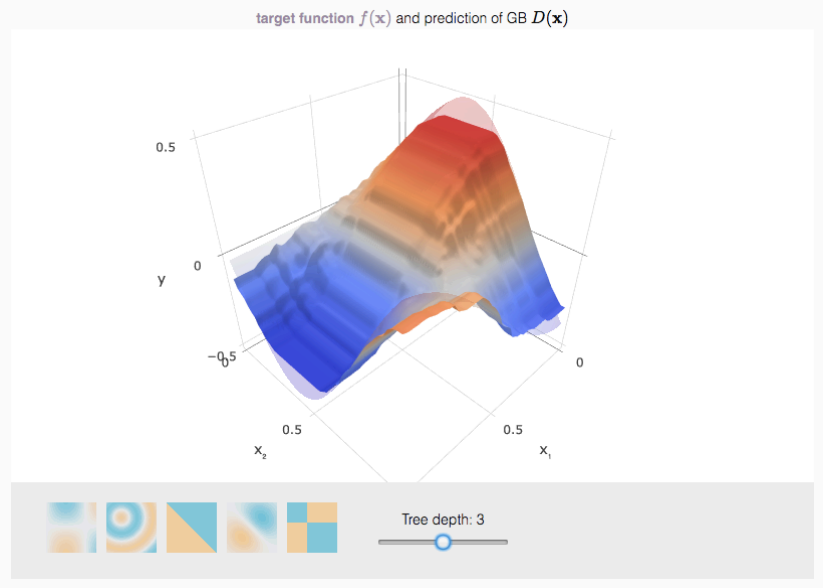
Justification is provided for which model appears to be the best to use given computational cost, model performance, and the characteristics of the data.

Eu concordo com sua escolha de Gradient Boosting aqui! É um dos melhores estimadores para este projeto. Vale a pena notar que pode não ser justo comparar modelos diferentes com seus parâmetros padrão. Por exemplo, as árvores de decisão tendem a não ter um bom desempenho com seus parâmetros padrão. Em geral, estimadores baseados em árvores tem obtido melhores resultados neste projeto. Uma possível explicação para isso é talvez a necessidade de estimadores com fronteiras de decisão não lineares para separação de classes.

Student is able to clearly and concisely describe how the optimal model works in layman's terms to someone who is not familiar with machine learning nor has a technical background.

Bônus

Este artigo Gradient Boosting explained é excelente para visualização deste estimador:



The final model chosen is correctly tuned using grid search with at least one parameter using at least three settings. If the model does not need any parameter tuning it is explicitly stated with reasonable justification.

Student reports the accuracy and F1 score of the optimized, unoptimized, models correctly in the table provided. Student compares the final model results to previous results obtained.

Ótima melhora no modelo usando GridSearch! AdaBoost e Gradient Boosting são geralmente os estimadores com melhores pontuações para este projeto.

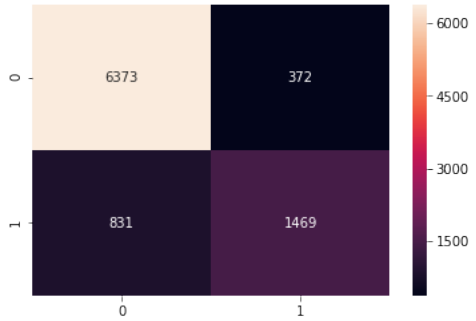
Bônus

Você também pode verificar seus resultados utilizando confusion matrix:

```
import seaborn as sns # Install using 'pip install seaborn'
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
%matplotlib inline

cm_test = confusion_matrix(y_test, best_clf.predict(X_test))

plt.figure(figsize=(7,5))
sns.heatmap(cm_test, annot=True, cmap='Greys', xticklabels=['No', 'Yes'], y
ticklabels=['No', 'Yes'])
plt.title('Confusion Matrix for the Test Set')
plt.ylabel('True')
plt.xlabel('Predicted')
```



Feature Importance

Student ranks five features which they believe to be the most relevant for predicting an individual's' income. Discussion is provided for why these features were chosen.

Student correctly implements a supervised learning model that makes use of the `feature_importances_` attribute. Additionally, student discusses the differences or similarities between the features they considered relevant and the reported relevant features.

Comentário

É válido observar que cada estimador com `feature_importances_` pode retornar diferentes top atributos dependendo da implementação do algoritmo interno.

Sugestão

Você também pode usar o atributo `feature_importances_` de `best_clf` já que ele já foi calibrado tendo assim uma melhor escolha dos top 5 atributos:

```
importances = best_clf.feature_importances_
```

Bonus

Além da importância dos atributos, uma alternativa é usar o cálculo da significância de cada atributo usando os pesos de um modelo de regressão logística (LR). A significância é calculada dividindo-se os coeficientes de LR pelos seus erro padrão (também conhecidos como Z-score). Você pode conferir mais detalhes no Capítulo 4.4.2 de The Elements of Statistical Learning . De acordo com a referência, um Z-score maior que 2 é significativo no nível de 5%.

Student analyzes the final model's performance when only the top 5 features are used and compares this performance to the optimized model from Question 5.

Comentário

Uma estratégia alternativa para seleção de atributos é usar técnicas de redução de dimensionalidade (PCA, por exemplo). Você verá mais detalhes sobre o PCA no próximo módulo.

BAIXAR PROJETO

RETORNAR