PCS5024 Atividade 3

Previsão de receita de um norte-americano baseado na base adult

Aluno: Filipe Assis Mourão

Número Usp : 8988914

Neste terceiro exercício foi pedido para se continuar à análise da base de dados “Adult Census Income” utilizando outros algoritmos.

Tal base de dados possui 15 colunas, 14 sendo features como idade, nível de escolaridade e estado civil, além de uma coluna dizendo se o descrito cidadão americano ganhava mais ou menos de 50 mil dólares anualmente.

A base de dados está amplamente disponível no site de competições kaggle, onde também é possível discutir possíveis soluções para o problema. As melhores acurácias para este problema variam entre 84% e 88% . Para se obter o melhor resultado possível, foi necessário primeiro um pré-processamento dos dados.

Inicialmente notou-se que a base possuía dados faltantes em 3 das 14 colunas, eram estas “workclass” que aparecia em cerca de 5.64% das linhas, “occupation” que aparecia em cerca de 5.66% das linhas e “native.country” que aparecia em cerca de 1.79% das linhas. Para simplificar o problema foi decidido que todas as linhas que possuíam dados faltantes seriam excluídas do problema, nisso o número total de 32561 linhas se reduziu para 30162 uma perda de 7.4% dos dados disponíveis.

Após à remoção de dados faltantes, foram feitas duas mudanças recomendadas por soluções que obtiveram uma alta acurácia no kaggle, a primeira é alterar à coluna de status matrimonial para casado ou solteiro, ao invés de uma das 7 possíveis possibilidades inicias. A segunda foi retirar à coluna de educação na forma numerada, pois representava os mesmos dados da coluna de educação. Finalmente, às colunas “income” e “sex” foram convertidas em colunas com dados binários.

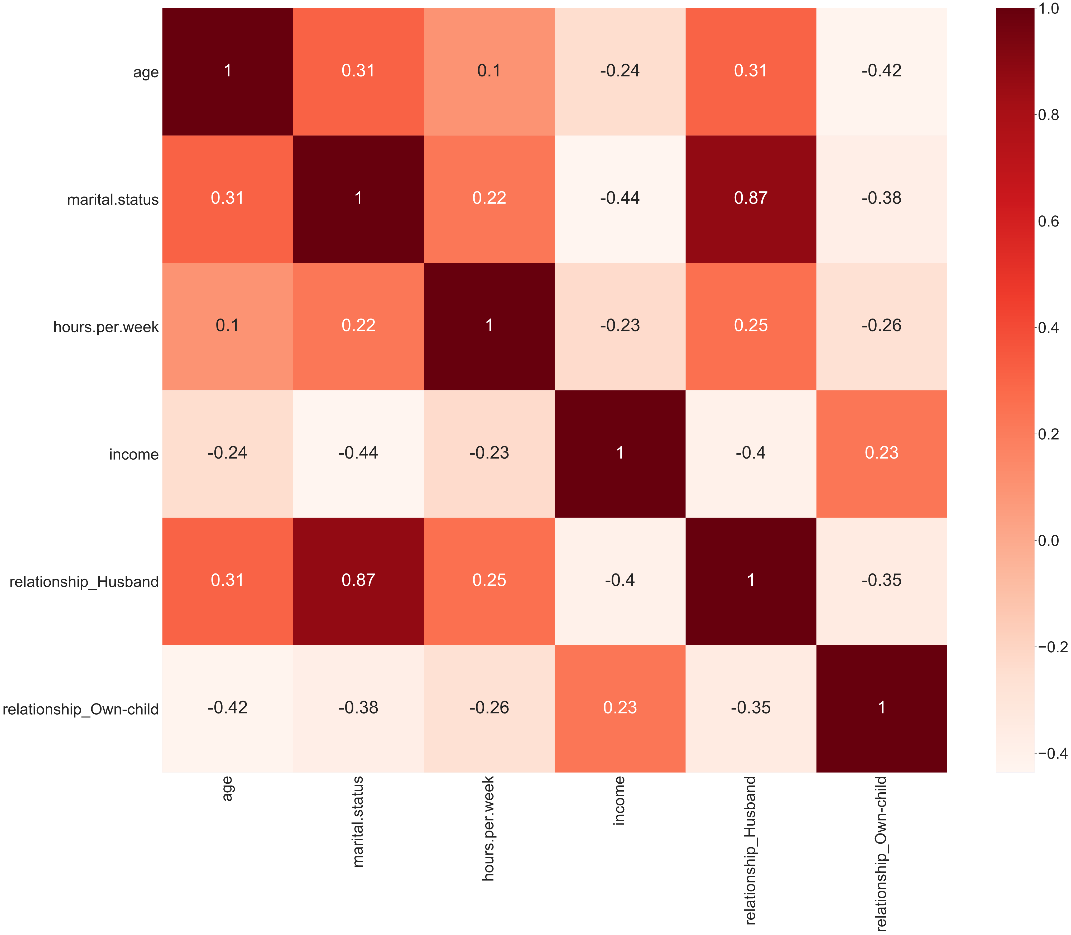
Após isso, foi notado que 6 colunas possuíam valores discretos, dessa forma, foi necessário decompor essas colunas em colunas com indicadores, através da função get\_dummies() da biblioteca pandas.

Em seguida, foi feita uma normalização entre os valores 0 e 1 em todas as 5 colunas com valores contínuos, utilizando a função preprocessing.MinMaxScaler() da biblioteca sklearn, para que a ordem de grandeza dos valores não influenciasse na construção do nosso modelo classificador.

Após a decomposição e normalização foi feita um mapa de calor com a correlação entre cada uma das colunas e a coluna “income” com o objetivo de escolher as colunas com maior relevância para criar o modelo classificador. Um mapa de calor contendo as 5 variáveis mais significativas é apresentado abaixo.

Pela correlação, percebe-se que as variáveis mais indicativas de renda são o status civil do entrevistado e se a pessoa entrevistada é o marido do casamento.

Foram então finalmente criados diferentes modelos utilizando um diferente número de variáveis significativas (3,5,10, 20 e 30), os dados foram divididos em 60% para treinamento, 20% para validação cruzada e 20% para teste.



**Imagem 1**: Mapa de calor com a correlação das 5 variáveis mais relevantes

Para cada algoritmo utilizado foi feito um hyperparameter tuining para se obter o melhor resultado possível. A descrição dos parâmetros otimizados em cada algoritmo é feita a seguir:

* KNN: Número de vizinhos
* Naive bayes: Tipo de algoritmo utilizado (gaussiano, bernoulli ou multinominal)
* Logistic Regression: A constante C que indica o quão forte será à regularização
* Decision Tree: Tamanho máximo da árvore
* Random Forest: Número de árvores utilizadas e tamanho máximo da árvore

Os 3 melhores resultados de cada algoritmo utilizando os dados de validação cruzada podem ser vistos na tabela abaixo:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Algorithm** | **number of relevant variables** | **accuracy** | **precision** | **recall** | **f1-score** |
| Random Forest | 30 | 0,858776728 | 0,867788462 | 0,957559682 | 0,910466583 |
| Random Forest | 30 | 0,858610973 | 0,867467467 | 0,957780725 | 0,910389747 |
| Random Forest | 20 | 0,858776728 | 0,869119421 | 0,955570292 | 0,910296905 |
| Logistic Regression | 30 | 0,846842367 | 0,867647059 | 0,938992042 | 0,901910828 |
| Logistic Regression | 30 | 0,846179347 | 0,866938776 | 0,938992042 | 0,901528014 |
| Logistic Regression | 30 | 0,845847837 | 0,866585067 | 0,938992042 | 0,901336728 |
| KNN | 30 | 0,837725841 | 0,870119023 | 0,921087533 | 0,894878127 |
| KNN | 30 | 0,835902536 | 0,86417972 | 0,92683466 | 0,894411263 |
| KNN | 20 | 0,835073761 | 0,861059955 | 0,930150309 | 0,89427266 |
| Naive bayes | 20 | 0,831758661 | 0,86728072 | 0,915782493 | 0,890871949 |
| Naive bayes | 10 | 0,812033814 | 0,825710992 | 0,949823165 | 0,883429276 |
| Decision Tree | 20 | 0,82115034 | 0,871949903 | 0,892572944 | 0,882140907 |
| Decision Tree | 20 | 0,820984585 | 0,872727273 | 0,891246684 | 0,881889764 |
| Decision Tree | 20 | 0,820321565 | 0,872133276 | 0,891025641 | 0,881478242 |
| Naive bayes | 20 | 0,820984585 | 0,881310895 | 0,879752431 | 0,880530973 |

**Imagem 2**: Top 3 resultados obtidos por cada algoritmo ordenados em maior f1-score

Pelo resultado acima, percebe-se que o algoritmo com o melhor resultado foi o random forest com os hiper-parâmetros maxima\_profundidade = 10 numero\_de\_arvores = 1100 e 30 variáveis significativas.

Foi então realizado um novo treinamento utilizando os dados de treinamento e validação cruzada e se obteve os seguintes valores de performance:

* Accuracy: 0.852
* Precision; 0.867
* Recall: 0.948
* F1-score: 0.906

O que se aproxima muito dos melhores resultados obtidos no kaggle utilizando algoritmos mais complexos como o xgboost.