Aprendizado Supervisionado

Eduardo Ouriques, Fernando Lima

Introdução

Este trabalho apresenta uma análise de performance dos modelos KNN, Naive Bayes e Árvore de decisão. Portanto, foi elaborado uma breve análise exploratória do dataset, e foram realizados treinamentos e testes dos modelos com o propósito de buscar uma melhor acurácia.

Análise do Dataset

O dataset contém dados de tipos de coberturas florestais, distribuídos em 581002 linhas e 55 colunas.

As colunas são divididas em:

- *"Elevação"*: Elevação em metros
- *"Aspecto"*: Aspecto em graus azimute
- *"Inclinação"*: Inclinação em graus
- *"Horizontal_Distance_To_Hydrology"*: Distância horizontal para recursos de água superficial mais próximos
- *"Vertical_Distance_To_Hydrology"*: Distância vertical para recursos de água de superfície mais próximos
- *"Horizontal_Distance_To_Roadways"*: Distância horizontal para a estrada mais próxima
- *"Hillshade_9am"* (índice de 0 a 255): Índice de Hillshade às 9h, solstício de verão
- *"Hillshade_Noon"* (índice de 0 a 255): Índice de Hillshade ao meio-dia, solstício de verão
- *"Hillshade_3pm"* (índice de 0 a 255): Índice de Hillshade às 3pm, solstício de verão

- *“Horizontal_Distance_To_Fire_Points”*: Distância horizontal para os pontos de ignição de incêndios florestais mais próximos

- *“Wilderness_Area”* (4 colunas binárias, 0 = ausência ou 1 = presença): Designação da área geográfica

- *“Soil_Type”* (40 colunas binárias, 0 = ausência ou 1 = presença): Designação do tipo de solo

- *“Cover_Type”* (7 tipos, inteiros 1 a 7): Designação do tipo de cobertura de floresta

Através do comando “head”, podemos ver uma amostra dos dados em questão, representados pela imagem a seguir:

```
In [14]: dataset = pd.read_csv("dataset.csv")
         dataset.head()
```

Out[14]:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_No
0	2596	51	3	258	0	510	221	2
1	2804	139	9	268	65	3180	234	2
2	2785	155	18	242	118	3090	238	2
3	2595	45	2	153	-1	391	220	2
4	2579	132	6	300	-15	67	230	2

Imagem 1 - Colunas do dataset

Como o número de colunas é extenso, o uso do scroll horizontal se faz necessário para a visualização completa da amostragem.

Através do comando “*dataset.dtypes*”, podemos ver que todos os dados são numéricos, como mostra a imagem a seguir:

```
In [9]: dataset.dtypes

Out[9]: Elevation                           int64
        Aspect                              int64
        Slope                               int64
        Horizontal_Distance_To_Hydrology    int64
        Vertical_Distance_To_Hydrology      int64
        Horizontal_Distance_To_Roadways     int64
        Hillshade_9am                       int64
        Hillshade_Noon                      int64
        Hillshade_3pm                       int64
        Horizontal_Distance_To_Fire_Points  int64
        Wilderness_Area1                    int64
        Wilderness_Area2                    int64
        Wilderness_Area3                    int64
        Wilderness_Area4                    int64
        Soil_Type1                          int64
        Soil_Type2                          int64
        Soil_Type3                          int64
        Soil_Type4                          int64
        Soil_Type5                          int64
        Soil_Type6                          int64
        Soil_Type7                          int64
        Soil_Type8                          int64
        Soil_Type9                          int64
        Soil_Type10                         int64
        Soil_Type11                         int64
        Soil_Type12                         int64
        Soil_Type13                         int64
        Soil_Type14                         int64
        Soil_Type15                         int64
        Soil_Type16                         int64
        Soil_Type17                         int64
        Soil_Type18                         int64
        Soil_Type19                         int64
        Soil_Type20                         int64
        Soil_Type21                         int64
        Soil_Type22                         int64
        Soil_Type23                         int64
        Soil_Type24                         int64
        Soil_Type25                         int64
        Soil_Type26                         int64
        Soil_Type27                         int64
        Soil_Type28                         int64
        Soil_Type29                         int64
        Soil_Type30                         int64
        Soil_Type31                         int64
        Soil_Type32                         int64
        Soil_Type33                         int64
        Soil_Type34                         int64
        Soil_Type35                         int64
        Soil_Type36                         int64
        Soil_Type37                         int64
        Soil_Type38                         int64
        Soil_Type39                         int64
        Soil_Type40                         int64
        Cover_Type                          int64
        dtype: object
```

Imagem 2 - Tipo das Colunas do dataset

Através do comando “*dataset.describe(include='all').transpose()*”, podemos ver uma análise mais descritiva do dataset. A imagem a seguir apresenta dados como quantidade, média, desvio padrão, mínimo, máximo e intervalo inter quartílicos (IQR).

```
In [19]: dataset.describe(include='all').transpose()
```

Out[19]:

	count	mean	std	min	25%	50%	75%	max
Elevation	581002.0	2959.371136	279.980764	1859.0	2809.0	2996.0	3163.0	3858.0
Aspect	581002.0	155.657158	111.913616	0.0	58.0	127.0	260.0	360.0
Slope	581002.0	14.103702	7.488241	0.0	9.0	13.0	18.0	66.0
Horizontal_Distance_To_Hydrology	581002.0	269.429680	212.549971	0.0	108.0	218.0	384.0	1397.0
Vertical_Distance_To_Hydrology	581002.0	46.419222	58.295524	-173.0	7.0	30.0	69.0	601.0
Horizontal_Distance_To_Roadways	581002.0	2350.165252	1559.257261	0.0	1106.0	1997.0	3328.0	7117.0
Hillshade_9am	581002.0	212.145838	26.769947	0.0	198.0	218.0	231.0	254.0
Hillshade_Noon	581002.0	223.318806	19.768789	0.0	213.0	226.0	237.0	254.0
Hillshade_3pm	581002.0	142.528609	38.274526	0.0	119.0	143.0	168.0	254.0
Horizontal_Distance_To_Fire_Points	581002.0	1980.292908	1324.176031	0.0	1024.0	1710.0	2550.0	7173.0
Wilderness_Area1	581002.0	0.448869	0.497379	0.0	0.0	0.0	1.0	1.0
Wilderness_Area2	581002.0	0.051435	0.220884	0.0	0.0	0.0	0.0	1.0
Wilderness_Area3	581002.0	0.436074	0.495897	0.0	0.0	0.0	1.0	1.0
Wilderness_Area4	581002.0	0.063621	0.244077	0.0	0.0	0.0	0.0	1.0
Soil_Type1	581002.0	0.005217	0.072039	0.0	0.0	0.0	0.0	1.0
Soil_Type2	581002.0	0.012952	0.113067	0.0	0.0	0.0	0.0	1.0
Soil_Type3	581002.0	0.008298	0.090713	0.0	0.0	0.0	0.0	1.0
Soil_Type4	581002.0	0.021336	0.144500	0.0	0.0	0.0	0.0	1.0
Soil_Type5	581002.0	0.002749	0.052356	0.0	0.0	0.0	0.0	1.0
Soil_Type6	581002.0	0.011315	0.105768	0.0	0.0	0.0	0.0	1.0
Soil_Type7	581002.0	0.000181	0.013442	0.0	0.0	0.0	0.0	1.0
Soil_Type8	581002.0	0.000308	0.017550	0.0	0.0	0.0	0.0	1.0
Soil_Type9	581002.0	0.001974	0.044388	0.0	0.0	0.0	0.0	1.0
Soil_Type10	581002.0	0.056168	0.230247	0.0	0.0	0.0	0.0	1.0
Soil_Type11	581002.0	0.021356	0.144589	0.0	0.0	0.0	0.0	1.0
Soil_Type12	581002.0	0.051585	0.221188	0.0	0.0	0.0	0.0	1.0
Soil_Type13	581002.0	0.030002	0.170592	0.0	0.0	0.0	0.0	1.0
Soil_Type14	581002.0	0.001029	0.032066	0.0	0.0	0.0	0.0	1.0
Soil_Type15	581002.0	0.000005	0.002272	0.0	0.0	0.0	0.0	1.0
Soil_Type16	581002.0	0.004897	0.069805	0.0	0.0	0.0	0.0	1.0
Soil_Type17	581002.0	0.005890	0.076519	0.0	0.0	0.0	0.0	1.0
Soil_Type18	581002.0	0.003268	0.057077	0.0	0.0	0.0	0.0	1.0
Soil_Type19	581002.0	0.006921	0.082903	0.0	0.0	0.0	0.0	1.0
Soil_Type20	581002.0	0.015936	0.125229	0.0	0.0	0.0	0.0	1.0
Soil_Type21	581002.0	0.001442	0.037951	0.0	0.0	0.0	0.0	1.0
Soil_Type22	581002.0	0.057440	0.232682	0.0	0.0	0.0	0.0	1.0
Soil_Type23	581002.0	0.099401	0.299200	0.0	0.0	0.0	0.0	1.0
Soil_Type24	581002.0	0.036623	0.187834	0.0	0.0	0.0	0.0	1.0
Soil_Type25	581002.0	0.000616	0.028551	0.0	0.0	0.0	0.0	1.0
Soil_Type26	581002.0	0.004456	0.066605	0.0	0.0	0.0	0.0	1.0
Soil_Type27	581002.0	0.001869	0.043194	0.0	0.0	0.0	0.0	1.0
Soil_Type28	581002.0	0.001628	0.040318	0.0	0.0	0.0	0.0	1.0
Soil_Type29	581002.0	0.198356	0.398762	0.0	0.0	0.0	0.0	1.0
Soil_Type30	581002.0	0.051928	0.221881	0.0	0.0	0.0	0.0	1.0
Soil_Type31	581002.0	0.044175	0.205485	0.0	0.0	0.0	0.0	1.0

Soil_Type32	581002.0	0.090394	0.286745	0.0	0.0	0.0	0.0	1.0
Soil_Type33	581002.0	0.077716	0.267724	0.0	0.0	0.0	0.0	1.0
Soil_Type34	581002.0	0.002773	0.052584	0.0	0.0	0.0	0.0	1.0
Soil_Type35	581002.0	0.003255	0.056957	0.0	0.0	0.0	0.0	1.0
Soil_Type36	581002.0	0.000205	0.014310	0.0	0.0	0.0	0.0	1.0
Soil_Type37	581002.0	0.000513	0.022642	0.0	0.0	0.0	0.0	1.0
Soil_Type38	581002.0	0.026602	0.161504	0.0	0.0	0.0	0.0	1.0
Soil_Type39	581002.0	0.023762	0.152308	0.0	0.0	0.0	0.0	1.0
Soil_Type40	581002.0	0.015060	0.121792	0.0	0.0	0.0	0.0	1.0
Cover_Type	581002.0	2.051444	1.396483	1.0	1.0	2.0	2.0	7.0

Imagem 3 - Detalhamento das colunas do dataset

Através do comando abaixo podemos visualizar um gráfico dos tipos de coberturas e das quantidades dos mesmos:

```
In [12]: fig = plt.figure(figsize=(11,6))
         fig = dataset['Cover_Type'].value_counts().plot.barh()
         fig.set_title('Cover Types')
         fig.set_ylabel('Cover Types')
         fig.set_xlabel('Quantities')
```

```
Out[12]: Text(0.5, 0, 'Quantities')
```

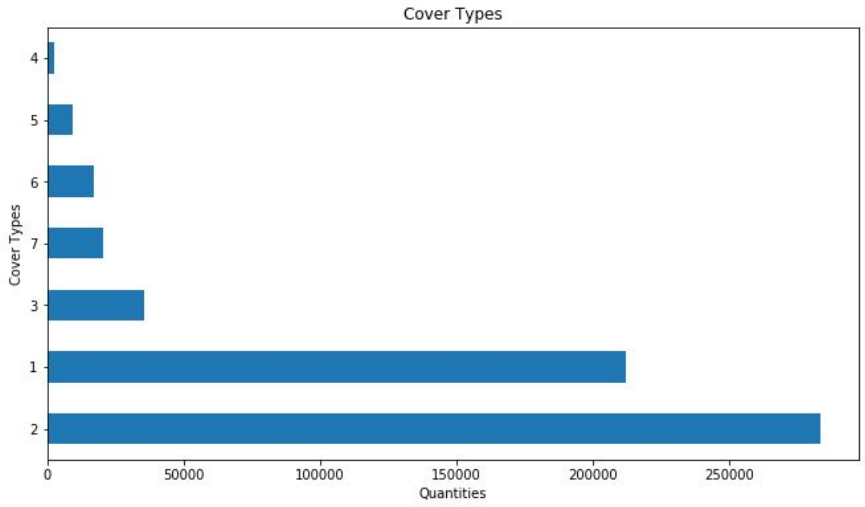

Imagem 4 - Quantidade por tipo

Concluímos que o tipo 1 e 2 são os predominantes do dataset.

Algoritmos

- KNN

    É um dos muitos algoritmos de aprendizagem supervisionada usado no campo de data mining e machine learning, ele é um classificador onde o aprendizado é baseado “no quão similar” é um dado (um vetor) do outro. O treinamento é formado por vetores de n dimensões.

    Com este modelo, utilizamos o classificador *KNeighborsClassifier* que representa KNN no pacote “sklearn.neighbors”, e utilizamos também o dataset citado na *Análise do Dataset*.

```
In [3]: print("Accuracy:", scores.mean())

Accuracy: 0.5414039569515248
```

Imagem 5 - Acurácia do KNN

Utilizando a configuração padrão, o resultado obtido foi uma acurácia de 54%, o que neste caso é resultado aceitável, porém não o melhor para ser usado.

Após feita a tunagem do algoritmo utilizando *Grid Search*, a acurácia do modelo aumentou em apenas 1% como valor “3” para o parâmetro “*n_neighbors’* que foi o valor que o *Grid Search* revelou como o melhor.

```
In [7]: knnmodel = KNeighborsClassifier()

        paramgrid = {'n_neighbors': [3,5,7,9]}

        knngridsearch = GridSearchCV(knnmodel, paramgrid, cv=10, scoring='accuracy')

        knngridsearch.fit(X, y)

Out[7]: GridSearchCV(cv=10, error_score='raise-deprecating',
               estimator=KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                   metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                   weights='uniform'),
               fit_params=None, iid='warn', n_jobs=None,
               param_grid={'n_neighbors': [3, 5, 7, 9]}, pre_dispatch='2*n_jobs',
               refit=True, return_train_score='warn', scoring='accuracy',
               verbose=0)

In [8]: knngridsearch.best_params_

Out[8]: {'n_neighbors': 3}

In [10]: knngridsearch.best_score_

Out[10]: 0.558683102639922

In [ ]:
```

Imagem 6 - Acurácia do KNN com hyper-tunning

- Naive Bayes

Este algoritmo é um classificador probabilístico baseado no "*Teorema de Bayes*", o qual foi criado por Thomas Bayes (1701 - 1761) para tentar provar a existência de Deus. [1]. Ao utilizar este algoritmo temos a característica de não correlação entre as features do dataset, o que neste caso, pode ser um fator crucial no resultado.

Com este modelo, utilizamos o classificador *GaussianNB* que representa Naive Bayes no pacote "sklearn.naive_bayes", e utilizamos também o dataset citado na *Análise do Dataset.*

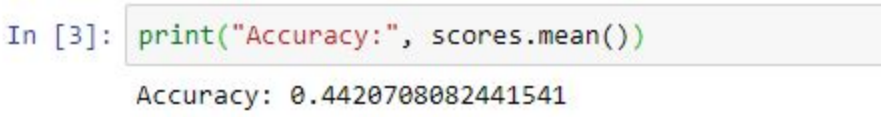

Imagem 7 - Acurácia do Naive Bayes

Utilizando a configuração padrão, o resultado obtido foi uma acurácia de 44%, o que neste caso é ruim.

- Árvores de Decisão

Árvores de decisão são métodos de aprendizado de máquinas supervisionado não-paramétricos, muito utilizados em tarefas de classificação e regressão.

Uma das principais vantagens das árvores de decisão é sua capacidade de atribuir valores específicos a problemas, decisões e resultados de cada decisão. Isso reduz a ambigüidade na tomada de decisões.

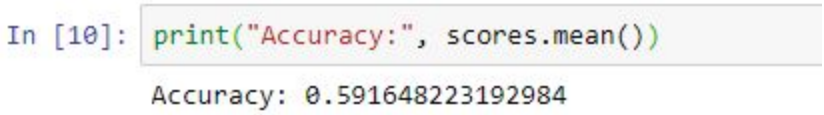

Imagem 8 - Acurácia da Árvore de decisão

Utilizando a configuração padrão, o resultado obtido foi uma acurácia de 59%, o que nos leva a concluir que, de fato, este é um algoritmo mais apropriado para o nosso objetivo.

Teste do Modelo

Este capítulo apresenta as imagens de como fizemos para carregar o modelo, e então, testarmos com o arquivo *"to_predict.csv"*

Exportação do modelo:

```
In [39]: from sklearn.externals import joblib

         filename = 'model.joblib'
         joblib.dump(model, filename)

Out[39]: ['model.joblib']
```

Resultado da predição com o modelo final:

```
In [45]: loaded_model = joblib.load(filename)

         dataset = pd.read_csv("to_predict.csv")

         result = loaded_model.predict(dataset[dataset.columns.values])

         print("Result:", result)

         Result: [2 5 1 6 5 5 2 3 4 6]
```

Conclusão

Após treinar e testar o nosso dataset com cada modelo, KNN, Naive Bayes e Árvore de decisão, concluímos que a árvore de decisão foi a que apresentou melhor desempenho. Para chegarmos a esta conclusão, verificamos a acurácia de cada modelo, e com isto obtivemos um resultado de 59%.

Referência

- [1] *Algoritmo de Classificação Naive Bayes* *https://www.organicadigital.com/seeds/algoritmo-de-classificacao-naive-bayes/*