DecisionTree_1 (1)

August 19, 2022

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
data = pd.read_csv('horse.csv')
data.head()
```

[2]:
```
  surgery    age  hospital_number  rectal_temp  pulse  respiratory_rate  \
0      no  adult           530101         38.5   66.0              28.0
1     yes  adult           534817         39.2   88.0              20.0
2      no  adult           530334         38.3   40.0              24.0
3     yes  young          5290409         39.1  164.0              84.0
4      no  adult           530255         37.3  104.0              35.0

  temp_of_extremities peripheral_pulse mucous_membrane capillary_refill_time  \
0                cool          reduced             NaN            more_3_sec
1                 NaN              NaN   pale_cyanotic            less_3_sec
2              normal           normal       pale_pink            less_3_sec
3                cold           normal   dark_cyanotic            more_3_sec
4                 NaN              NaN   dark_cyanotic            more_3_sec

   … packed_cell_volume total_protein abdomo_appearance abdomo_protein  \
0  …               45.0           8.4               NaN            NaN
1  …               50.0          85.0            cloudy            2.0
2  …               33.0           6.7               NaN            NaN
3  …               48.0           7.2    serosanguious            5.3
4  …               74.0           7.4               NaN            NaN

      outcome  surgical_lesion lesion_1 lesion_2  lesion_3 cp_data
0        died               no    11300        0         0      no
1  euthanized               no     2208        0         0      no
2       lived               no        0        0         0     yes
3        died              yes     2208        0         0     yes
4        died               no     4300        0         0      no

[5 rows x 28 columns]
```

[3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 28 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   surgery                299 non-null    object
 1   age                    299 non-null    object
 2   hospital_number        299 non-null    int64
 3   rectal_temp            239 non-null    float64
 4   pulse                  275 non-null    float64
 5   respiratory_rate       241 non-null    float64
 6   temp_of_extremities    243 non-null    object
 7   peripheral_pulse       230 non-null    object
 8   mucous_membrane        252 non-null    object
 9   capillary_refill_time  267 non-null    object
 10  pain                   244 non-null    object
 11  peristalsis            255 non-null    object
 12  abdominal_distention   243 non-null    object
 13  nasogastric_tube       195 non-null    object
 14  nasogastric_reflux     193 non-null    object
 15  nasogastric_reflux_ph  53 non-null     float64
 16  rectal_exam_feces      197 non-null    object
 17  abdomen                181 non-null    object
 18  packed_cell_volume     270 non-null    float64
 19  total_protein          266 non-null    float64
 20  abdomo_appearance      134 non-null    object
 21  abdomo_protein         101 non-null    float64
 22  outcome                299 non-null    object
 23  surgical_lesion        299 non-null    object
 24  lesion_1               299 non-null    int64
 25  lesion_2               299 non-null    int64
 26  lesion_3               299 non-null    int64
 27  cp_data                299 non-null    object
dtypes: float64(7), int64(4), object(17)
memory usage: 65.5+ KB
```

[4]:
```
data.isna().sum()
```

[4]:
```
surgery                   0
age                       0
hospital_number           0
rectal_temp              60
pulse                    24
respiratory_rate         58
temp_of_extremities      56
```

```
peripheral_pulse          69
mucous_membrane           47
capillary_refill_time     32
pain                      55
peristalsis               44
abdominal_distention      56
nasogastric_tube         104
nasogastric_reflux       106
nasogastric_reflux_ph    246
rectal_exam_feces        102
abdomen                  118
packed_cell_volume        29
total_protein             33
abdomo_appearance        165
abdomo_protein           198
outcome                    0
surgical_lesion            0
lesion_1                   0
lesion_2                   0
lesion_3                   0
cp_data                    0
dtype: int64
```

[5]:
```
data.outcome.value_counts()
```

[5]:
```
lived         178
died           77
euthanized     44
Name: outcome, dtype: int64
```

[6]:
```
features = data.drop(['outcome'], axis = 1)
target = data[['outcome']]
```

[7]:
```
features.shape,target.shape
```

[7]:
```
((299, 27), (299, 1))
```

[8]:
```
features.dtypes
```

[8]:
```
surgery                    object
age                        object
hospital_number             int64
rectal_temp               float64
pulse                     float64
respiratory_rate          float64
temp_of_extremities        object
peripheral_pulse           object
```

```
mucous_membrane            object
capillary_refill_time      object
pain                       object
peristalsis                object
abdominal_distention       object
nasogastric_tube           object
nasogastric_reflux         object
nasogastric_reflux_ph     float64
rectal_exam_feces          object
abdomen                    object
packed_cell_volume        float64
total_protein             float64
abdomo_appearance          object
abdomo_protein            float64
surgical_lesion            object
lesion_1                    int64
lesion_2                    int64
lesion_3                    int64
cp_data                    object
dtype: object
```

[9]:
```
features_transformed = pd.get_dummies(features)
```

[10]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

[11]:
```
X_train , X_test, y_train, y_test = train_test_split(features_transformed,␣
 ↪target, random_state = 10)
```

[12]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(224, 67)
(75, 67)
(224, 1)
(75, 1)
```

[13]:
```
from sklearn.impute import SimpleImputer
```

[14]:
```
imputer = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
```

[15]:
```
X_train = imputer.fit_transform(X_train)
X_test = imputer.fit_transform(X_test)
```

0.1 Decision Tree

[36]:
```
my_DT_model = DecisionTreeClassifier(max_depth=7)
```

[37]:
```
my_DT_model.fit(X_train,y_train)
```

[37]:
```
DecisionTreeClassifier(max_depth=7)
```

[38]:
```
my_preds = my_DT_model.predict(X_test)
```

[39]:
```
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
```

[40]:
```
accuracy_score(y_test, my_preds)
```

[40]:
```
0.6266666666666667
```

[41]:
```
print(confusion_matrix(y_test, my_preds))
```

```
[[ 6  3  6]
 [ 5  4  2]
 [ 4  8 37]]
```

[42]:
```
print(classification_report(y_test, my_preds))
```

```
              precision    recall  f1-score   support

        died       0.40      0.40      0.40        15
  euthanized       0.27      0.36      0.31        11
       lived       0.82      0.76      0.79        49

    accuracy                           0.63        75
   macro avg       0.50      0.51      0.50        75
weighted avg       0.66      0.63      0.64        75
```

[50]:
```
features.columns
```

[50]:
```
Index(['surgery', 'age', 'hospital_number', 'rectal_temp', 'pulse',
       'respiratory_rate', 'temp_of_extremities', 'peripheral_pulse',
       'mucous_membrane', 'capillary_refill_time', 'pain', 'peristalsis',
       'abdominal_distention', 'nasogastric_tube', 'nasogastric_reflux',
       'nasogastric_reflux_ph', 'rectal_exam_feces', 'abdomen',
       'packed_cell_volume', 'total_protein', 'abdomo_appearance',
       'abdomo_protein', 'surgical_lesion', 'lesion_1', 'lesion_2', 'lesion_3',
       'cp_data'],
      dtype='object')
```

```
[53]: import operator
      sorted(list(zip(features.columns,my_DT_model.feature_importances_)),␣
       ↪key=operator.itemgetter(1), reverse=True)
```

```
[53]: [('respiratory_rate', 0.15811606733352418),
       ('mucous_membrane', 0.14269762191554108),
       ('temp_of_extremities', 0.13328172362507973),
       ('cp_data', 0.07225624381928748),
       ('hospital_number', 0.06361554996427964),
       ('rectal_exam_feces', 0.05922435066217908),
       ('rectal_temp', 0.0462423438344502),
       ('nasogastric_reflux_ph', 0.04267256707100241),
       ('surgical_lesion', 0.030073999650039797),
       ('surgery', 0.0297961156153894),
       ('peripheral_pulse', 0.029238610770872014),
       ('nasogastric_reflux', 0.027432364545644412),
       ('abdomen', 0.008223264663320757),
       ('age', 0.0),
       ('pulse', 0.0),
       ('capillary_refill_time', 0.0),
       ('pain', 0.0),
       ('peristalsis', 0.0),
       ('abdominal_distention', 0.0),
       ('nasogastric_tube', 0.0),
       ('packed_cell_volume', 0.0),
       ('total_protein', 0.0),
       ('abdomo_appearance', 0.0),
       ('abdomo_protein', 0.0),
       ('lesion_1', 0.0),
       ('lesion_2', 0.0),
       ('lesion_3', 0.0)]
```

0.1.1 Voting Classifier

```
[43]: from sklearn.ensemble import VotingClassifier, BaggingClassifier,␣
       ↪RandomForestClassifier
      from sklearn.linear_model import LogisticRegression
      from sklearn.svm import SVC
```

```
[44]: my_logreg_clf = LogisticRegression()
      rf_clf = RandomForestClassifier()
      svm_clf = SVC()
```

```
[45]: my_vt_clf = VotingClassifier(estimators=[('lr', my_logreg_clf), ('rf', rf_clf),␣
       ↪('svc', svm_clf)])
```

[46]:
```
my_vt_clf.fit(X_train, y_train)
```

```
/usr/local/lib/python3.7/site-packages/sklearn/utils/validation.py:63:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  return f(*args, **kwargs)
/usr/local/lib/python3.7/site-packages/sklearn/linear_model/_logistic.py:765:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

[46]:
```
VotingClassifier(estimators=[('lr', LogisticRegression()),
                             ('rf', RandomForestClassifier()), ('svc', SVC())])
```

[47]:
```
y_pred = my_vt_clf.predict(X_test)
accuracy_score(y_test, y_pred)
```

[47]:
```
0.6533333333333333
```

[ ]: