AppliedMLFinal_nguyen

December 9, 2020

0.1 Intro

This final assignment is similar to the midterm, but certain parts are different so please read the new instructions carefully to make sure you don't miss anything.

In this class, we have worked to understand both general principles of machine learning methods and specific details of particular machine learning techniques. For this project, you will apply what you have learned in this course to formulate and answer your own question, using ML.

Important: Be sure to include code and answers in the correct cells of the notebook. Otherwise we might miss your answers.

Please use this notebook to turn in your work.

Points: 25

You will be graded based on: 1. **Technical completeness**. *(15 points)*

Did you meet the technical requirements for the project? For instance, did you describe hyperparameter tuning and include plots, where necessary.

2. **Creativity and imagination**. *(5 points)*

Did you form an interesting, original and creative question, and explain why it is important to answer that question?

3. **Presentation**. *(5 points)*

Did you do a good job presenting your results? Would your notebook make sense to a person who was not familiar with your project? You should take time to write clearly, simplify your code and explain what you are doing in your notebook. Make sure your plots are well-labeled and appropriately scaled.

0.2 Original question [instructions]

Unlike in the midterm project, for the final project you will be asked to formulate and answer an original question. An original question is a question that you come up with on your own, rather than a question that is formulated for you. For instance, downloading a dataset from Kaggle and predicting the labels provided in the dataset is not asking an original question because Kaggle contributors have already defined the task for you. As before, you should take time to explain why this question matters and how it fits with existing understanding of your topic.

If you are stuck, here are some strategies for finding original questions:

1. You might take an existing analysis and offer a new twist. For instance, you might start with a dataset used for predicting the winners of NBA games and instead cluster the data to see if you can identify blowouts (when one a team wins by a lot).
2. You might compare datasets in new ways. For instance, you might test if features that are good for predicting NBA games are also good for predicting NFL games, and then investigate why you might observe a difference.
3. You might apply your own knowledge of a subject to ask a new question. For instance, if you know a lot about minor league baseball, this might help you ask interesting and original questions that others have not thought of before.
4. Modify or expand existing work. You might start with a blog post or paper that shares data, and then extend or modify the existing study.
5. Systematically analyze errors in an existing analysis. You might start with a blog post or paper that shares data, and then revisit and rexamine some of the questions with empirical tests using code to see if the author overlooked certain angles or drew unsubstantiated conclusions.

0.3 Related work [required 5604, optional 4604 for 3 points extra credit]

One way to find an original question is to start by analyzing and considering related work. Using this cell, please summarize two research papers related to your question (one paragraph per paper). Be sure to link to the paper and list the title, authors and venue so that others can understand your work. For each paper, you should explain (1) what the authors found and (2) how their work informs your own question or approach.

Paper 1

Paper 2

0.4 Original question [answer]

Please write a clear and concise explanation of your original question

This is a major part of the assignment. Please take time to carefully explain what is original about your question.

Can machine learning predict if an NBA team will make the playoffs? With analytics taking a larger role in basketball, I thought the application of machine learning techniques on NBA data would make for an interesting project. While machine learning projects exist that determine the probability a team wins against a certain opponent, this analysis is not very useful for regular season games as top teams move more and more toward load management to save star players for the playoffs by limiting their use in regular season games. With load management becoming more prevalent in the NBA, simple analysis of individual regular season games becomes less valuable. Instead, my project looks to classify potential playoff teams which then can be applied to existing NBA machine learning projects as a way to identify teams that may wish to save their stars for the playoffs.

0.5 Midterm project expansion

You can expand your midterm project for the final project, but your final and midterm must be substantially different. For instance, if you expand your midterm, you must ask a new question, try new models, use new data, perform new error analysis, etc. If you are expanding your midterm *project, you must explain the differences between your midterm and final below. Err on the side of caution; make sure your two projects are substantially different.*

If your final is not an expansion of your midterm, then you can skip this part.

N/A

0.6 Data (writing)

Using this cell, please write a short, clear paragraph explaining what data you will use to answer your question. You do not need to go gather custom datasets for this class, although you are welcome to do so. Your data gathering must allow you to ask an original question, but you are not required to construct a custom data set. If applicable, describe how you plan to split between the training and test sets.

The data that I use in this project comes from ESPN which maintains records of NBA team stats each season. For simplicity, I have limited my analysis to the 2019 and 2020 NBA seasons. The dataset I have created includes the following per game variables: points, field goals made, field goals attempted, field goal percentage, 3-pointers made, 3-pointers attempted, 3-point percentage, free throws made, free throws attempted, free throw percentage, offensive rating, defensive rating, rebounds, assists, steals, blocks, turnovers, and personal fouls. Unfortunately, the data does not include a variable for if the associated team made the playoffs and I had to hardcode that data into a spreadsheet. I will perform a 70/30 split of the data with sklearn's train_test_split function. The function will randomly split 70% of the data into a test set and 30% of the data into a training set.

[2]:
```
## Code for data preprocessing

# Include your code to load, clean and split data in this cell. You must␣
↪complete this step in the project.

import pandas as pd
nba = pd.read_csv('nba.csv')

y = nba["PLAYOFF"]
x = nba.drop(["Team","PLAYOFF"], axis = 1)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.7)
```

0.7 Model selection and tuning

This section is similar to the midterm.

Using this cell, please write a short, clear paragraph explaining how you selected and tuned your model for the project. You must answer the following two questions in this cell (1) Why is your model an appropriate choice for your data? (2) What hyperparameters does your model have and how did you select them? (3) What features did you choose and why? You must engineer at least 1 feature

Since the target variable, did the team make the playoffs, is a binary variable, I chose to implement a Logistic Regression since the logistic function returns probabilities that the target class is 1 (true) or 0 (false). Sklearn's documentation of the LogisticRegression function includes many hyper parameters, but I will only discuss the two that I deem important for this project, penalty and C. The penalty parameter determines which regularization function is used to penalize overfitting. C determines the strength of that penalty. Additionally, I will use PCA to reduce the dimension of the input features. Although the dataset is not large, I wanted to test if a model with reduced dimensions can still return high accuracy.

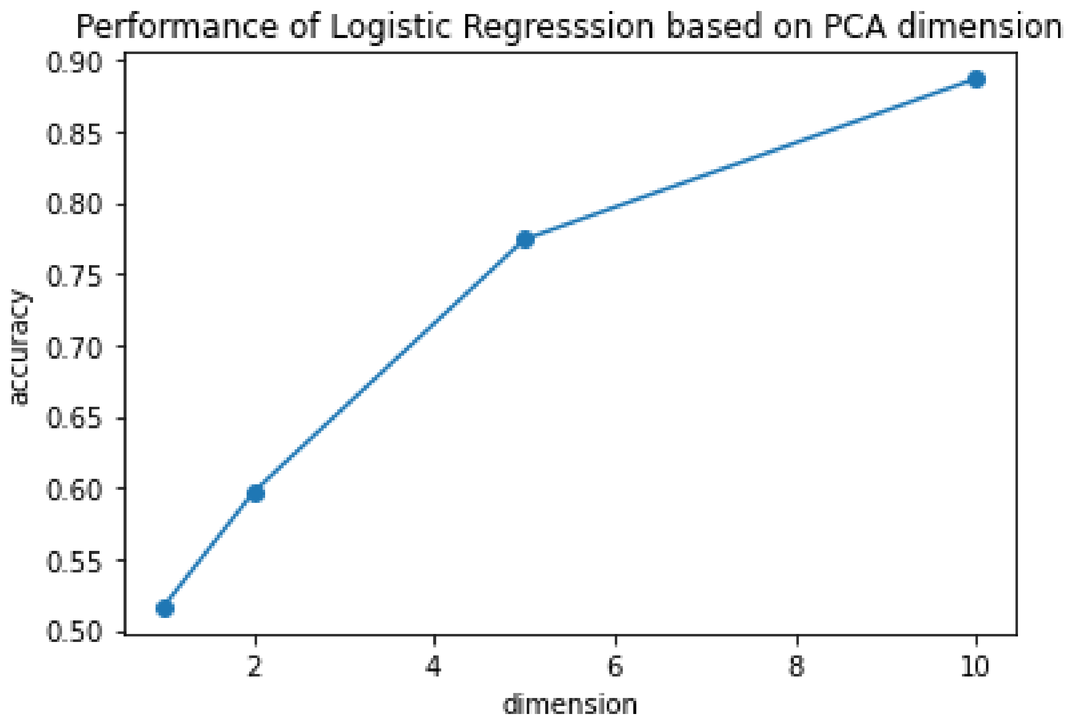
[28]:
```
## Code for model selection and tuning + feature engineering

# Include any code to select and tune your model in this cell. You must␣
↪complete this step in the project.

from sklearn.linear_model import LogisticRegression
from sklearn.decomposition import PCA
from sklearn.metrics import accuracy_score

c = [0.001, 0.01, 0.1, 1, 10, 100]
dim = [1, 2, 5, 10]
pen = ["l1", "l2"]

c_acc = []
dim_acc = []
pen_acc = []

# penalty tuning
for p in pen:
    logit = LogisticRegression(penalty = p, solver = "saga", max_iter = 10000)
    logit.fit(x_train, y_train)
    y_logit = logit.predict(x_train)
    acc = accuracy_score(y_train, y_logit)
    pen_acc.append(acc)
    print(p, ": ", acc)

# c tuning
print("\n")
for i in c:
    logit = LogisticRegression(C = i, max_iter = 10000)
    logit.fit(x_train, y_train)
    y_logit = logit.predict(x_train)
    acc = accuracy_score(y_train, y_logit)
    c_acc.append(acc)
    print(i, ": ", acc)

# PCA dimensionality reduction
print("\n")
```

```
for d in dim:
    pca = PCA(d)
    x_proj = pca.fit_transform(x_train)
    logit = LogisticRegression(max_iter = 10000)
    logit.fit(x_proj, y_train)
    y_logit = logit.predict(x_proj)
    acc = accuracy_score(y_train, y_logit)
    dim_acc.append(acc)
    print(d, ": ", acc)
```

```
l1 :  0.7580645161290323
l2 :  0.7580645161290323


0.001 :  0.7580645161290323
0.01 :  0.8064516129032258
0.1 :  0.8870967741935484
1 :  0.967741935483871
10 :  0.967741935483871
100 :  0.9838709677419355


1 :  0.5161290322580645
2 :  0.5967741935483871
5 :  0.7741935483870968
10 :  0.8870967741935484
```

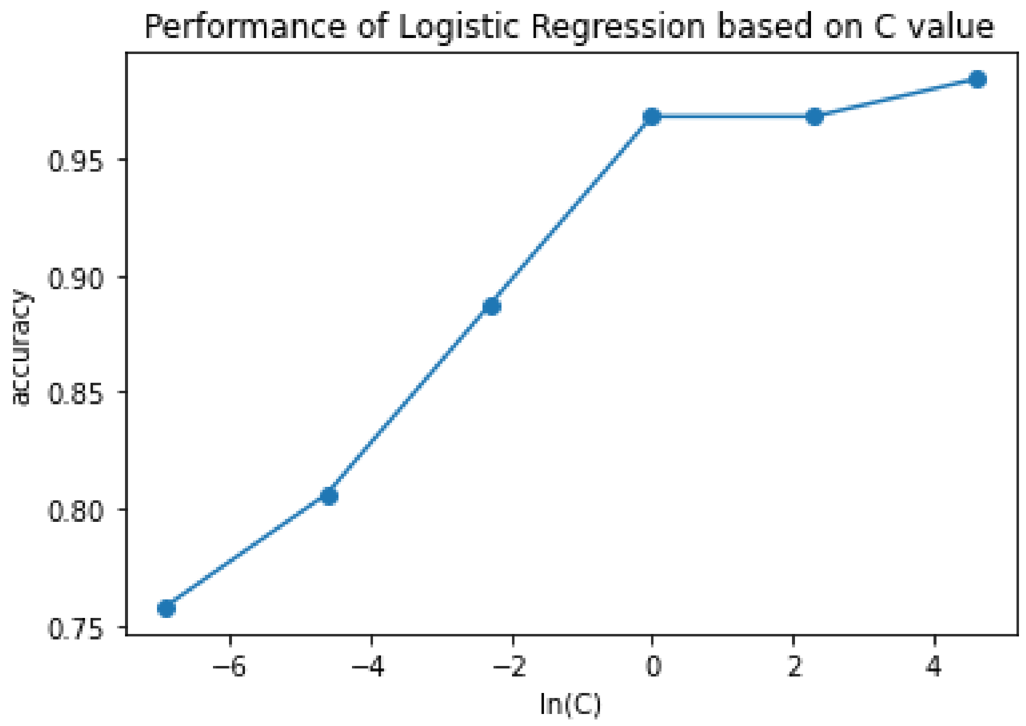
[29]:
```
## Plot or table

# Include a plot or table explaining how you selected and tuned your model. You␣
 ↪must complete this step in the project.

import matplotlib.pyplot as plt
import numpy as np

plt.scatter(dim, dim_acc)
plt.plot(dim, dim_acc)
plt.title("Performance of Logistic Regresssion based on PCA dimension")
plt.ylabel("accuracy")
plt.xlabel("dimension")
```

[29]:
```
Text(0.5, 0, 'dimension')
```



```
[30]: plt.scatter(np.log(c), c_acc)
      plt.plot(np.log(c), c_acc)
      plt.title("Performance of Logistic Regression based on C value")
      plt.ylabel("accuracy")
      plt.xlabel("ln(C)")
```

```
[30]: Text(0.5, 0, 'ln(C)')
```



0.8 Results

This section is similar to the midterm.

Using this cell, please write a short, clear paragraph explaining your results. In this class, we have mostly focused on accuracy. It is OK to measure your results in another quantitative way (e.g. precision or likelihood). Whatever you pick, make sure you are clear on what you are doing, and make sure you explain why your measurement of success makes sense.

The two metrics I used to evaluate the performance of my model are accuracy and cross-validation score. The table below reports these metrics. The model returned 75% accuracy when predicting if an NBA team made the playoffs in a season. Although this result is not as high as I would have liked, correctly predicting the 3/4 of the test data is still not a bad result for the limited team data provided. The leave-one-out cross validation score of the model was only 60.7%, slightly higher than randomly guessing if a team made the playoffs.

[33]:
```
## Code

# Include code showing how you arrived at your results. You must complete this␣
 ↪step in the project.
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import LeaveOneOut

pca = PCA(10)
x_proj = pca.fit_transform(x_train)
```

```
x_proj_test = pca.fit_transform(x_test)

logit = LogisticRegression(penalty = "l2", C = 100, max_iter = 10000)

logit.fit(x_proj, y_train)
y_predict = logit.predict(x_proj_test)
scores = cross_val_score(logit, x_proj_test, y_test, cv=LeaveOneOut())
```

[34]:
```
## Plot or table

# Include a plot or table explaining your results. You must complete this step␣
↪in the project.
print("                 Model Performance                 ")
print("---------------------------------------------------")
print("Accuracy:", accuracy_score(y_predict, y_test))
print("Leave one out cross validation:", scores.mean())
```

```
                 Model Performance
---------------------------------------------------
Accuracy: 0.75
Leave one out cross validation: 0.6071428571428571
```

0.9 Error analysis

Please note the added instructions below. Some students made mistakes on error analysis for the midterm.

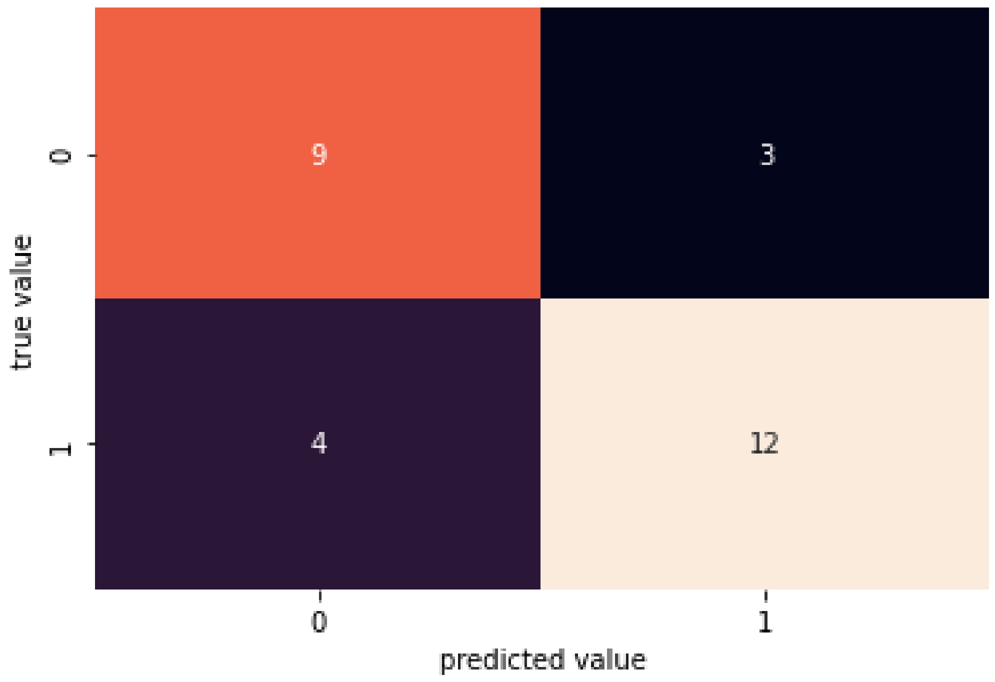
Using this cell, please write a short, clear paragraph explaining what errors your model seems to be making, and offer a brief explanation based on your code below.

To get full credit for error analysis, it is not sufficient to simply report the performance of your model. You must analyze your model's output and form theories about why the model might be failing. Ideally, you should also test those theories by writing code or looking at examples to understand what is going on. To analyze errors, you might make a confusion matrix, sample misclassified data or analyze learned weights. Try to be systematic in understanding errors.

It seems that my model frequently predicted that a playoff team would not qualify for the playoffs, making this mistake four times as can be seen in the following confusion matrix. Since the whole dataset only covered two seasons. Also, out of 12 non-playoff teams, the model incorrectly classified 3 of them. I suspect that since I only used basic per game stats, the model was not able to correctly classify playoff teams, with high accuracy. It is worth noting that 75% of non-playoff teams and 75% of playoff teams were correctly classified, indicating that the model performs about equally when determining if a team won't make the playoffs and if a team will make the playoffs.

[35]:
```
## Error analysis

# Include code for error analysis here, to justify your conclusions.
# You might make a confusion matrix, sample misclassified data, analyze learned␣
 ↪weights, or use any other method
# discussed in class, or which makes sense for your model

from sklearn.metrics import confusion_matrix
import seaborn as sns

mat = confusion_matrix(y_test, y_predict)

sns.heatmap(mat, annot = True, cbar = False)
plt.ylabel("true value")
plt.xlabel("predicted value")
```

```
[35]: Text(0.5, 15.0, 'predicted value')
```



0.10 Discussion

5604 students be sure to describe how your findings relate to conclusions from prior work

Using this cell, please write a short, clear paragraph describing how your results answer or do not answer your question. What new questions arise from your work?

As I mentioned in the error analysis section, I chose to only use basic NBA team stats since advanced team stats were not readily availiable as downloadable datasets. Limiting my project to basic stats may have negatively affected the performance of my model. Additional work in this area should include advanced NBA team stats which includes valuable metrics such as pace, true shooting percentage, player impact estimate, and effective field goal percentage. While the model did not perform as well as I would have liked, I won't abandon machine learning as a method to predict playoff teams. I believe machine learning can accurately predict playoff teams when given the right data. I refuse to believe that the 75% accuracy of this model is merely coincidental.