
Machine learning Team_1

11기 원혜진

13기 기다연

13기 박무성

13기 박주영

13기 배은지



Machine learning Team_1

- 001 전처리
- 002 사용 모델
- 003 Test 결과
- 004 데이터 경진 대회를 통해 배운 점



Part 1.

EDA 및 데이터 전처리 작업



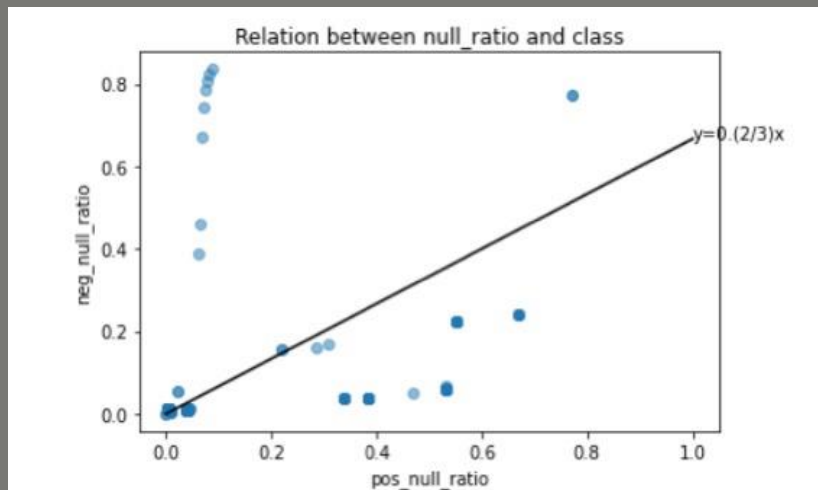
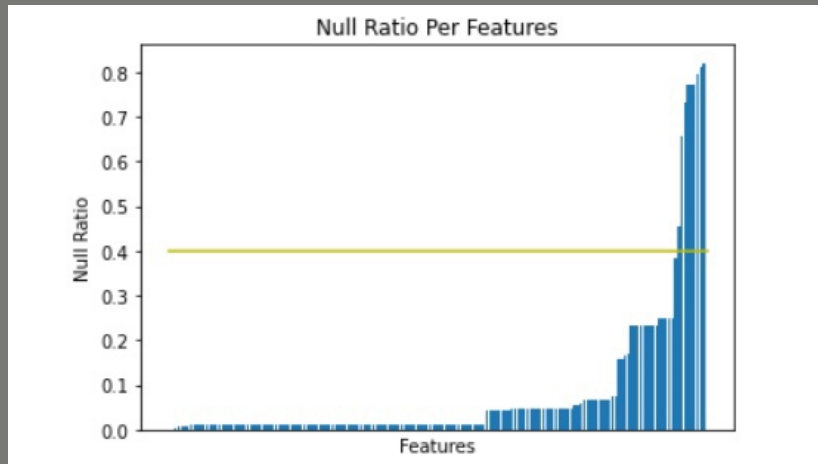
EDA 및 데이터 전처리 작업

“머신러닝 1조, 자신감을 가지다 ”

```
train_data = Data("./Train_data.csv", mode="Train")
# mode: Train / Test
dropped, POS_MEDIAN, TOTAL_MEDIAN = train_data.processNA()
# na 많은 컬럼 drop, na값 대체
train_data.replaceHisto()
# 히스토그램 대표값 대체
train_scaler = train_data.standardize()
# 정규화 - !!!classBalancing 함수 이전 실행 필수_SMOTE 이전 정규화 필요하기 때문!!!
train_data.classBalancing()
# SMOTE 사용한 class balancing
to_drop = train_data.featureSelection(isFitted=False, to_drop=None)
pca = train_data.dimReduction()
# PCA 사용 차원 축소
```

- Fir, 결측치 처리
- Sec, 표준화 및 불균형 자료 해결
- Thir, 특성 변수 해결
- Fou, PCA를 사용한차원 축소

First, 결측치 처리



결측치 비율 40% 이상 Drop



“결측치가 있다는것 그 자체가 정비가
필요하다는 지표가 될 수 있지 않을까?”



Class가 pos일때 유독 결측치가 많다...?
: pos의 median으로 대체하자!

First, 결측치 처리

```
def processNA(self, dropped=None, pos_median=None, total_median=None):
    if self.mode=="Train":
        # check null columns per cats
        pos_data = self.src_data[self.src_data["class"]==1]
        neg_data = self.src_data[self.src_data["class"]==0]

        pos_df, neg_df = pos_data.isnull().sum().to_frame(name="pos_null_cnt"), neg_data.isnull().sum().to_frame(name="neg_null_cnt")
        null_check = pos_df.join(neg_df)
        null_check["total_null"] = null_check["pos_null_cnt"] + null_check["neg_null_cnt"]
        null_check.insert(1, "pos_cnt", len(pos_data))
        null_check.insert(3, "neg_cnt", len(neg_data))

        null_check["pos_null_ratio"] = null_check.pos_null_cnt/null_check.pos_cnt
        null_check["neg_null_ratio"] = null_check.neg_null_cnt/null_check.neg_cnt
        null_check["total_null_ratio"] = (null_check.pos_null_cnt + null_check.neg_null_cnt)/len(self.src_data)

        # drop cols w/ too many na's
        dropped = null_check[null_check.total_null_ratio>=0.4]
        dropped = set(dropped.index)
        self.src_data.drop(dropped, inplace=True, axis=1)

        # na replacement
        with_pos = null_check[null_check.pos_null_ratio>=0.3]
        with_pos = set(with_pos.index) - dropped
        self.POS_MEDIAN = self.src_data[with_pos].median(axis=0).to_dict()

        with_total = null_check[null_check.pos_null_ratio<0.3]
        with_total = set(with_total.index) - dropped
        self.TOTAL_MEDIAN = self.src_data[with_total].median(axis=0).to_dict()

    elif self.mode=="Test":
        if dropped is None or pos_median is None or total_median is None:
            print("\n:::ERROR:::kwargs cannot be None in the Test mode")
            return -2
        else:
            # drop cols w/ too many na's
            self.src_data.drop(dropped, inplace=True, axis=1)

            # na replacement
            self.POS_MEDIAN = pos_median
            self.TOTAL_MEDIAN = total_median

    self.src_data.fillna(value=self.POS_MEDIAN, inplace=True)
    self.src_data.fillna(value=self.TOTAL_MEDIAN, inplace=True)

    print(f"Dropped {len(dropped)} columns: {' '.join(dropped)}")
    print(f"Replaced with POS_MEDIAN and TOTAL_MEDIAN")

    if self.mode=="Train": self.X = self.src_data.iloc[:, 1:]
    else: self.X = self.src_data

    return dropped, self.POS_MEDIAN, self.TOTAL_MEDIAN
```

결측치 비율 40% 이상 Drop

9개 특성 변수 제거

“결측치가 있다는것 그 자체가 정비가
필요하다는 지표가 될 수 있지 않을까?”

(null & pos) / pos 비율이 30% 이상이면
Missing Value를 pos_median으로 대체

Second, 표준화 및 불균형 자료 해결

```
def standardize(self, scaler = None):
    if self.mode=='Train':
        scaler = StandardScaler()
        scaler.fit(self.X)
    else:
        if scaler is None:
            print("#n::ERROR::scaler cannot be None in the Test mode")
            return -3
        scaler = scaler

    scaled_X = scaler.transform(self.X)
    scaled_X = pd.DataFrame(scaled_X, columns=self.X.columns, index=self.X.index)
    self.X = scaled_X

    print(f"Standardization finished")
    print(self.X.describe().applymap("{0:.2f}".format))
    return scaler
```



표준화 진행

```
def classBalancing(self, rndn_state=42):
    print('Original dataset shape: %s' % Counter(self.y))

    sm = SMOTE(random_state=rndn_state)
    X_res, y_res = sm.fit_resample(self.X, self.y)
    print('Resampled dataset shape: %s' % Counter(y_res))

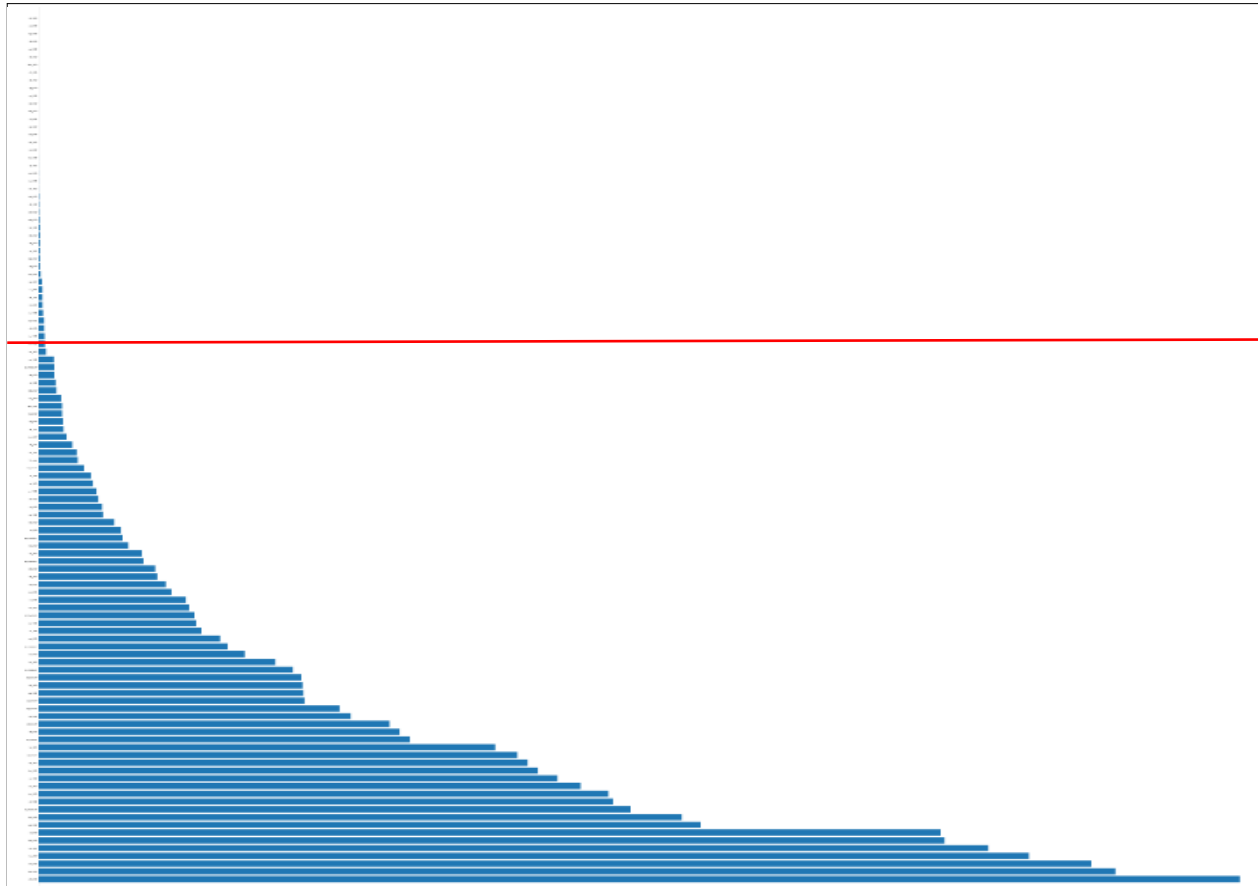
    scaler = StandardScaler()
    scaled_X_res = scaler.fit_transform(X_res)
    scaled_X_res = pd.DataFrame(scaled_X_res, columns=X_res.columns, index=X_res.index)

    print(scaled_X_res.describe())
    self.X, self.y = scaled_X_res, y_res
    print('Oversampling finished')
```



SMOTE 적용

Third, 중요도가 떨어지는 특성 변수 제거



기준치 0.0001

: 총 34개 특성 변수 제거

Fourth, PCA를 사용해 차원 축소

```
#PCA 방법 1: 전체 분산의 95% 설명하는 최소 개수 선택
def dimReduction(self, pca_fitted = None):
    if self.mode=='Train':
        pca = PCA()
        pca.fit(self.X)

        cumsum_ex_var = np.cumsum(pca.explained_variance_ratio_)
        dim = np.argmax(cumsum_ex_var >= 0.95) + 1
        print("number of dimensions: " + str(dim))

        plt.figure(figsize=(5, 5))
        plt.xlabel('Number of components')
        plt.ylabel('cumulative sum of explained variance')
        plt.plot(range(1, cumsum_ex_var.shape[0] + 1), cumsum_ex_var)

        pca = PCA(n_components=dim) #
        pca.fit(self.X)

        print("explained variance ratio: " + str(sum(pca.explained_variance_ratio_)))
    else :
        if pca_fitted is None:
            print("#n:::ERROR:::pca cannot be None in the Test mode#n")
            return -4
        pca = pca_fitted

    X_reduced = pca.transform(self.X)
    self.X = pd.DataFrame(X_reduced)
    print('PCA finished')
    return pca
```

전체의 95%를 설명

: 보수적으로 잡았지만 전체 변수들의 20%만 생존

```
#PCA 방법 2: 고유값이 1보다 큰 주성분 선택
def dimReduction(self, pca_fitted = None):
    if self.mode=='Train':
        X_features = self.X.values.T
        covariance_matrix = np.cov(X_features)
        eig_vals, eig_vecs = np.linalg.eig(covariance_matrix)
        dim = len(eig_vals[eig_vals > 1]) # 고유값이 1보다 큰 주성분의 개수
        print("number of dimensions: " + str(dim))

        pca = PCA(n_components=dim)
        pca.fit(self.X)

        print("explained variance ratio: " + str(sum(pca.explained_variance_ratio_)))
    else :
        if pca_fitted is None:
            print("#n:::ERROR:::pca cannot be None in the Test mode#n")
            return -4
        pca = pca_fitted

    X_reduced = pca.transform(self.X)
    self.X = pd.DataFrame(X_reduced)
    print('PCA finished')
    return pca
```

Eigen Value > 1

Part 2.

사용 Model 및 자체 Test 결과



사용 모델 및 자체 Test 결과

○ Support Vector Machine

- linear
- polynomial

○ Random Forest

○ Logistic Regression

```
[[10927, 260],  
 [ 302, 10885]]
```

```
[[10964 223]  
 [ 29 11158]]
```

RF Loss : 16730

```
[[10743 444]  
 [ 294 10893]]
```

Random Forest

- Eigen Value > 1

95%를 설명



“우리 1등하는거 아니야...?”

기다연
확실하진 않지만,,, 저번에 우승한 팀이 loss가 19900인가 했다는데,,
ㅎㅎ,, 그 팀도 이긴거겠조?!!
오전 8:24

박주영
헐
오전 8:24


기다연
상품 물어보러 갈께요,,ㅎ
오전 8:24

박주영
말로만 하던 1등이 진짜 우리에게,,?
오전 8:24

기다연
ㅋㅋㅋㅋㅋㅋㅋㅋ ㅋㅋㅋ ㅋㅋ
ㅋㅋㅋㅋㅋㅋ
오전 8:24

박주영
ㅋㅋㅋㅋㅋㅋ 좋은걸로 준비해달
라고 부탁드립니다요 ㅎ
오전 8:24

기다연
조은걸로 부탁드립니다 올께요 ~_~
오전 8:25

기다연

오전 9:07

기다연
(유감스럽게도 상품은 아직 정해지
지 않았네요..)
오전 9:09

박주영
아쉬운 소식이네요,,
오전 9:23

원혜진
내가 1등이다
오후 10:51

“우리 꼴찌 하는거 아니야...?”



“회장님이 실수했을거야.”



18 안수빈

“0이 하나 더 있네...?”



accuracy: 56,189%
loss: 138120입니다

오후 1:16



박주영

근데 학습데이터의 결과랑 이렇게
차이날 수 있는건가요..?

오후 1:58



기다연

그러게요TTTT 너무 놀랐어요T

오후 1:58



박주영

회장님이 실수하신거면 좋겠다,,

오후 1:58



“열의 제거가 정보의 손실을 가져온건 아닐까?”



“feature selection를 제외하고 전처리해보자!”



나는, 그 실수가 아프다

머신러닝 분반장 명재성님

아

방금까지 코드 봤는데

오후 7:21

머신러닝 분반장 명재성님

왜그런지 대충 알거같은게

real test 말고 만드신 test에도

오후 7:22

smote가 적용됐네요?

오후 7:22

전처리가

오후 7:22

같이 되어있어서

그런거죠....?

오후 7:22

머신러닝 분반장 명재성님

그런거 같습니다 제생각에는

오후 7:22

감사합니다...!

오후 7:22

```
train_data = Data("./Train_data.csv", mode="Train")
# mode: Train / Test
dropped, POS_MEDIAN, TOTAL_MEDIAN = train_data.processNA()
# na 많은 칼럼 drop, na값 대체
train_data.replaceHisto()
# 히스토그램 대표값 대체
train_scaler = train_data.standardize()
# 정규화 - !!!classBalancing 함수 이전 실행 필수 SMOTE 이전 정규화 필요하기 때문!!!
train_data.classBalancing()
# SMOTE 사용한 class balancing
to_drop = train_data.featureSelection(isFitted=False, to_drop=None)
pca = train_data.dimReduction()
# PCA 사용 차원 축소

X, y = train_data.X, train_data.y
```

- Test data에도 Smote를 적용
- 예측해야할 자료는 불균형 문제가 해결되지 않은 상태일텐데 이를 간과

Part 3.

What we learn from Data analysis contest



감사합니다

