Projet de la Pédiction Conforme

odjejulesgeraud

November 2024

1 Introduction

Ce rapport présente l'implémentation et l'analyse de trois approches complémentaires en apprentissage automatique :

- Une méthode de prédiction conforme appliquée à un problème de régression sur le dataset California Housing, permettant d'obtenir des intervalles de prédiction fiables pour l'estimation des prix immobiliers.
- Une extension de la prédiction conforme au cas de la classification multiclasse, illustrée sur le dataset Iris, démontrant la capacité à fournir des ensembles de prédiction garantis pour la classification d'espèces.
- Une implémentation de la régression quantile sur des données synthétiques, offrant une approche alternative pour la modélisation de l'incertitude à travers l'estimation de différents quantiles de la distribution conditionnelle.

Ces trois méthodes offrent des perspectives complémentaires sur la quantification de l'incertitude en apprentissage automatique, allant de la régression à la classification, en passant par l'estimation de distributions complètes.

2 Implémentation de la Prédiction Conforme Application au California Housing Dataset

2.1 Choix et Analyse du Dataset

2.1.1 Présentation du Dataset

Le California Housing Dataset a été choisi pour notre implémentation de prédiction conforme. Les données comprennent 20,640 observations avec 8 variables prédictives et 1 variable cible (PRICE).

2.1.2 Variables du dataset

— `MedInc` : Revenu médian du bloc
— `HouseAge` : Âge médian des maisons du bloc
— `AveRooms` : Nombre moyen de pièces
— `AveBedrms` : Nombre moyen de chambres
— `Population` : Population du bloc
— `AveOccup` : Nombre moyen d'occupants
— `Latitude` : Latitude géographique
— `Longitude` : Longitude géographique
— `PRICE` (target) : Valeur médiane des maisons du bloc

2.2 Pertinence pour la Prédiction Conforme

Ce dataset est particulièrement adapté pour notre étude car :
— Il présente un problème de régression réel
— Les prix immobiliers ont une variabilité naturelle qui justifie l'utilisation d'intervalles de prédiction
— Les données sont propres et ne nécessitent pas de prétraitement complexe

2.3 Implémentation de la Prédiction Conforme

2.3.1 Structure du Code

Notre implémentation s'articule autour d'une classe principale `ConformePredictionProject` avec les composants suivants :

2.3.2 Initialisation et Chargement des Données

```
class ConformePredictionProject:
    def __init__(self, alpha=0.1):
        self.housing = fetch_california_housing()
        self.alpha = alpha
        self.df = pd.DataFrame(
            self.housing.data,
            columns=self.housing.feature_names
        )
        self.df['PRICE'] = self.housing.target
```

2.3.3 Préparation des Données

```python
def prepare_data(self, test_size=0.2, calib_size=0.25):
    X = self.df.drop('PRICE', axis=1)
    y = self.df['PRICE']

    # Splits des donn es
    X_temp, X_test, y_temp, y_test = train_test_split(
        X, y, test_size=test_size
    )
    X_train, X_calib, y_train, y_calib = train_test_split(
        X_temp, y_temp, test_size=calib_size
    )
```

2.4 Résultats et Analyse

2.4.1 Métriques de Performance

Les résultats obtenus sont les suivants :
— Couverture : 89.9% (objectif : 90%)
— Largeur moyenne des intervalles : 1.654
— MSE : 0.295
— MAE : 0.365

2.4.2 Analyse des Performances

La couverture empirique de 89.9% est remarquablement proche de la valeur cible de 90%, démontrant une excellente calibration du modèle. Les erreurs (MSE et MAE) sont raisonnables compte tenu de la complexité du domaine immobilier.

2.5 Discussion

2.5.1 Points Forts

— Code modulaire et bien structuré
— Processus de calibration robuste
— Visualisations intégrées pour l'analyse

2.5.2 Limitations

— Intervalles de largeur fixe
— Pas d'adaptation à l'hétéroscédasticité
— Dépendance à la qualité du RandomForest

2.6 Conclusion

L'implémentation réalisée atteint ses objectifs avec une excellente calibration et des intervalles de prédiction fiables.

3 Prédiction Conforme pour la Classification Multiclasse
Application au Dataset Iris

3.1 Présentation du Dataset

3.1.1 Description Générale

Le dataset Iris comprend 150 observations avec 4 caractéristiques mesurées pour trois espèces d'iris différentes. Les caractéristiques sont :

- Longueur du sépale (cm)
- Largeur du sépale (cm)
- Longueur du pétale (cm)
- Largeur du pétale (cm)

3.1.2 Distribution des Données

La distribution des classes est parfaitement équilibrée :

Classe	Nombre d'échantillons	Pourcentage
0	50	33.33%
1	50	33.33%
2	50	33.33%

TABLE 1 – Distribution des classes dans le dataset Iris

3.2 Implémentation

3.2.1 Configuration du Modèle

Le modèle de base utilisé est un Random Forest Classifier avec les paramètres suivants :

```
RandomForestClassifier(
    n_estimators=100,
    max_depth=5,
    random_state=42
```

```
)
```

3.2.2 Préparation des Données

Les données ont été divisées selon le schéma suivant :
— Ensemble d'entraînement : 60%
— Ensemble de calibration : 20%
— Ensemble de test : 20%

3.2.3 Calcul des Scores de Conformité

```python
def compute_conformity_scores(self, model, X_calib,
    y_calib):
     proba_pred = model.predict_proba(X_calib)
     conformity_scores = []
     for i, y_true in enumerate(y_calib):
         conformity_scores.append(1 - proba_pred[i, y_true])
     return np.array(conformity_scores)
```

3.3 Résultats et Analyse

3.3.1 Métriques de Performance

Les résultats obtenus sont les suivants :
— **Couverture** : 93.3%
— **Taille moyenne des ensembles** : 1.000
— **Exactitude** : 93.3%

3.3.2 Interprétation des Résultats

La couverture de 93.3% dépasse légèrement le niveau de confiance visé (90%), indiquant une calibration légèrement conservative. La taille moyenne des ensembles de 1.0 suggère que le modèle est très discriminant et que les classes sont bien séparables.

3.4 Discussion

3.4.1 Points Forts

— Excellence des performances (93.3% d'exactitude)
— Calibration robuste
— Prédictions décisives (ensembles de taille 1)

3.4.2 Limitations

— Dataset relativement simple
— Classes parfaitement équilibrées
— Nombre limité d'observations

3.5 Conclusion

L'implémentation de la prédiction conforme sur le dataset Iris démontre l'efficacité de cette approche pour la classification multiclasse. Les résultats excellents en termes de couverture et de précision, combinés à des ensembles de prédiction de taille optimale, confirment la pertinence de la méthode.

4 Analyse et Implémentation de la Régression Quantile

4.1 Objectif

L'objectif de cette étude était d'implémenter une régression quantile permettant d'estimer différents niveaux de la distribution conditionnelle de la variable cible. Cette approche offre une modélisation plus complète de l'incertitude par rapport aux méthodes de régression classiques.

4.2 Données Utilisées

Les données synthétiques ont été générées avec les caractéristiques suivantes :
— 1000 échantillons
— Variable d'entrée $X \sim \mathcal{N}(0, 1)$
— Relation non-linéaire avec hétéroscédasticité
— Bruit dépendant de X : $\epsilon \sim \mathcal{N}(0, 0.1 + 0.3|X|)$

4.3 Méthodologie

4.3.1 Configuration du Modèle

```
class QuantileRegression:
    def __init__(self, quantiles=[0.05, 0.5, 0.95]):
        self.quantiles = quantiles
        self.models = {}
```

4.3.2 Paramètres du Modèle

Les hyperparamètres du GradientBoostingRegressor ont été configurés comme suit :

— loss = 'quantile'
— n_estimators = 100
— max_depth = 4
— learning_rate = 0.1

4.4 Analyse des Résultats

4.4.1 Métriques de Couverture

Quantile	Couverture Observée	Objectif	Écart
5%	7.5%	5%	+2.5%
50%	53.0%	50%	+3.0%
95%	94.0%	95%	-1.0%

Tableau 2 – Résultats de couverture par quantile

4.4.2 Interprétation Détaillée

Quantile 5% La couverture observée de 7.5% indique une estimation légèrement conservative du quantile inférieur, avec une surestimation du risque de 2.5 points de pourcentage.

Quantile 50% La médiane estimée présente une excellente précision avec une couverture de 53%, très proche de l'objectif théorique de 50%.

Quantile 95% Le quantile supérieur montre une légère sous-estimation avec une couverture de 94%, soit un point de moins que l'objectif de 95%.

4.5 Points Forts et Limitations

4.5.1 Points Forts

— Précision des estimations, particulièrement pour la médiane
— Gestion efficace de l'hétéroscédasticité
— Capture des relations non-linéaires

4.5.2 Limitations

— Nécessité d'entraîner un modèle par quantile

— Coût computationnel croissant avec le nombre de quantiles
— Léger biais dans les quantiles extrêmes

4.6 Conclusion

L'implémentation de la régression quantile a démontré une capacité satisfaisante à estimer les différents niveaux de la distribution conditionnelle. Les résultats montrent une précision particulièrement bonne pour la médiane, avec des déviations mineures mais acceptables pour les quantiles extrêmes.

A Code d'Évaluation de la régression quantile

```python
def evaluate_quantile_predictions(y_true, predictions, quantiles):
    metrics = {}
    for q in quantiles:
        coverage = np.mean(y_true <= predictions[q])
        metrics[f'coverage_{q}'] = coverage
    return metrics
```

B Expressions Mathématiques

La fonction de perte pour la régression quantile est donnée par :

$$\rho_\tau(u) = u(\tau - 1\{u < 0\})$$

où τ est le quantile désiré et u est l'erreur de prédiction.

C Code Source Complet

Voir le fichier python en copie de ce rapport.